

Derwent WPI

(c) 2009 Thomson Reuters. All rights reserved.

0011215737

WPI Acc no: 2002-154792/

Related WPI Acc No: 2001-451199; 2002-114606; 2002-122312; 2002-122326; 2003-661705; 2004-313572

XRFX Acc No: N2002-117667

**Control apparatus for extensible computing system in a dynamically sized, highly scaleable and available server using virtual server farms created from wide scale computing fabric**

Patent Assignee: TERRASPRING INC (TERR-N)

Inventor: AZIZ A; GRAY M; MARKSON T; PATTERSON M

Patent Family: 7 patents, 96 countries

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
WO 2002003203	A2	20020110	WO 2001US19053	A	20010613	200220	B
AU 200171307	A	20020114	AU 200171307	A	20010613	200237	E
EP 1323037	A2	20030702	EP 2001950298	A	20010613	200344	E
			WO 2001US19053	A	20010613		
US 6597956	B1	20030722	US 1999150394	P	19990823	200354	E
			US 2000502170	A	20000211		
			US 2000213090	P	20000620		
			US 2000630440	A	20000802		
TW 535064	A	20030601	TW 2001114836	A	20010619	200374	E
JP 2004508616	W	20040318	WO 2001US19053	A	20010613	200420	E
			JP 2002508204	A	20010613		
AU 2001271307	A8	20051013	AU 2001271307	A	20010613	200611	E

Priority Applications (no., kind, date): US 1999150394 P 19990823; US 2000502170 A 20000211; US 2000213090 P 20000620; US 2000630440 A 20000802

#### Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes
WO 2002003203	A2	EN	61	19	
National Designated States,Original	AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CO CR CU CZ DE DK DM DZ EC EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT TZ UA UG UZ VN YU ZA ZW				
Regional Designated States,Original	AT BE CH CY DE DK EA ES FI FR GB GH GM GR IE IT KE LS LU MC MW MZ NL OA PT SD SE SL SZ TR TZ UG ZW				
AU 200171307	A	EN			Based on OPI patent WO 2002003203
EP 1323037	A2	EN			PCT Application WO 2001US19053
					Based on OPI patent WO 2002003203

Regional Designated	AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL
---------------------	--

States,Original	PT RO SE SI TR					
US 6597956	B1		EN			Related to Provisional US 1999150394
						C-I-P of application US 2000502170
						Related to Provisional US 2000213090
TW 535064	A		ZH			
JP 2004508616	W		JA	107		PCT Application WO 2001US19053
						Based on OPI patent WO 2002003203
AU 2001271307	A8		EN			Based on OPI patent WO 2002003203

### Alerting Abstract WO A2

NOVELTY - A sample web site contains a single computing element or machine (100) with a central processor unit (102) and a disc (104) and is coupled to a global packet switched data network (106) or to another network, while a wide scale computing fabric is logically divided into virtual server farms for various organizations on demand. First subsets of processing and storage resources are selected and are communicatively coupled to each other.

DESCRIPTION - INDEPENDENT CLAIMS are included for a method of managing processing resources and for a computer readable medium with instructions.

USE - Controlling a computing grid.

DESCRIPTION OF DRAWINGS - The drawing shows a sample web site

100 Computing machine

102 Central processor unit

104 Disc

106 Data network

## Original Publication Data by Authority

### Original Abstracts:

Methods and apparatus providing, controlling and managing a dynamically sized, highly scalable and available server farm are disclosed. A Virtual Server Farm (VSF) is created out of a wide scale computing fabric ("Computing Grid") which is physically constructed once and then logically divided up into VSFs for various organizations on demand. Each organization retains independent administrative control of a VSF. A VSF is dynamically firewalled within the Computing Grid. Allocation and control of the elements in the VSF is performed by a control plane connected to all computing, networking, and storage elements in the computing grid through special control ports. The internal topology of each VSF is under control of the control plane. No physical rewiring is necessary in order to construct VSFs in many different configurations, including single-tier Web server or multi-tier Web-server, application server, database server configurations.

A Virtual Server Farm (VSF) is created out of a wide scale computing fabric ("Computing Grid") which is physically constructed once and then logically divided up into VSFs for various organizations on demand. Allocation and control of the elements in the VSF is performed by a control plane connected to all computing, networking, and storage elements in the computing grid through special control ports. The control plane is comprised of a control mechanism hierarchy that includes one or more master control process mechanisms communicatively coupled to one or more slave control process mechanisms. The one or more master control process mechanisms instruct the slave control process mechanisms to establish VSFs by selecting subsets of processing and storage resources.

Methods and apparatus providing, controlling and managing a dynamically sized, highly scalable and available server farm are disclosed. A Virtual Server Farm (VSF) is created out of a wide scale computing fabric ("Computing Grid") which is physically constructed once and then logically divided up into VSFs for various organizations on demand. Each organization retains independent administrative control of a VSF. A VSF is dynamically firewalled within the Computing Grid. Allocation and control of the elements in the VSF is performed by a control plane connected to all computing, networking, and storage elements in the computing grid through special control ports. The internal topology of each VSF is under control of the control plane. No physical rewiring is necessary in order to construct VSFs in many different configurations, including single-tier Web server or multi-tier Web-server, application server, database server configurations.

L'invention concerne des procedes et un appareil fournissant, commandant et gerant un groupe de serveurs dimensionne de maniere dynamique, hautement evolutif et disponible. Un groupe de serveurs virtuels (VSF) est cree a partir d'un programme de calcul a grande echelle ("grille de calcul") lequel est construit physiquement en une fois, puis divise logiquement en plusieurs VSF pour diverses organisations a la demande. Chaque organisation conserve la commande administrative independante d'un VSF. Un VSF est dote d'un pare-feu de facon dynamique a l'interieur de la grille de calcul. L'affectation et la commande des elements dans le VSF sont executees par un plan de commande connecte a tous les elements de calcul, de reseau et de stockage dans la grille de calcul par l'intermediaire de ports de commande speciaux. La topologie interne de chaque VSF est sous la commande du plan de commande. Aucun recablage physique n'est necessaire pour construire des VSF dans les nombreuses et differentes configurations, notamment des configurations de serveurs Web a simple niveau ou de serveurs Web multiniveau, de serveurs d'application et de serveurs de base de donnees.

Basic Derwent Week: 200220

Equivalent to Ref. 17

(19) 日本国特許庁(JP)

(12) 公表特許公報(A)

(11) 特許出願公表番号

特表2004-508616

(P2004-508616A)

(43) 公表日 平成16年3月18日(2004.3.18)

(51) Int. Cl. 7	F I	テーマコード (参考)
G06 F 15/16	G06 F 15/16 640 A	5B034
G06 F 9/46	G06 F 15/16 620 A	5B045
G06 F 11/20	G06 F 15/16 620 B	5B098
	G06 F 15/16 620 C	
	G06 F 9/46 360 C	
審査請求 未請求 予備審査請求 有 (全 107 頁) 最終頁に続く		
(21) 出願番号 特願2002-508204 (P2002-508204)	(71) 出願人 502062375	
(86) (22) 出願日 平成13年6月13日 (2001.6.13)	テラスブリッジ・インコーポレーテッド	
(85) 翻訳文提出日 平成14年12月17日 (2002.12.17)	アメリカ合衆国 カリフォルニア州 94	
(86) 国際出願番号 PCT/US2001/019053	538 フレモント ミルモント ドライ	
(87) 国際公開番号 W02002/003203	ブ 48800	
(87) 国際公開日 平成14年1月10日 (2002.1.10)	(74) 代理人 100086759	
(31) 優先権主張番号 60/213,080	弁理士 森田 喜平	
(32) 優先日 平成12年6月20日 (2000.6.20)	(72) 発明者 アズイス, アシャー	
(33) 優先権主張国 米国 (US)	アメリカ合衆国 カリフォルニア州 94	
(31) 優先権主張番号 09/630,440	555 フレモント タナガー コモン	
(32) 優先日 平成12年8月2日 (2000.8.2)	4180	
(33) 優先権主張国 米国 (US)	(72) 発明者 マークソン, トム	
	アメリカ合衆国 カリフォルニア州 94	
	402 サン マテオ マウンズ ロード	
	30	
最終頁に続く		

(54) 【発明の名称】 拡張可能コンピューティングシステムの制御方法および装置

## (57) 【要約】

動的に縮小拡大され、極めて拡張可能で利用可能なサーバファームを提供、制御、および管理する方法および装置が開示されている。仮想サーバファーム (VSF) は、物理的に構成された後で、要求があり次第、様々な組織のために VSF に論理的に分割される大規模コンピューティング構造 (「コンピューティンググリッド」) から生成される。各組織は、VSF の独立管理制御を維持する。VSF は、コンピューティンググリッド内で動的にファイアウォールされる。VSF における要素の割り当てと制御は、特殊制御ポートを介して、コンピューティンググリッド内の全てのコンピューティング、ネットワークング、および記憶要素に接続された制御プレーンによって行われる。各 VSF の内部トポロジーは、制御プレーンの制御下にある。単層ウェブサーバまたは多層ウェブサーバ、アプリケーションサーバ、データベースサーバの構成を含む多数の異なる構成において VSF を構成するために、物理的な配線のし直しは必要ない。

## 【特許請求の範囲】

## 【請求項1】

マスター制御機構と、  
マスター制御機構に通信接続され、且つマスター制御機構からの一又は二以上の命令に応

じて、  
処理リソースの集合から処理リソースの第1サブセットを選択し、  
記憶リソースの集合から記憶リソースの第1サブセットを選択し、且つ  
処理リソースの第1サブセットを記憶リソースの第1サブセットに通信接続させることによ

り、  
処理リソースの第1サブセットおよび記憶リソースの第1サブセットを含む第1論理リソ 10  
ースグループを確立するように構成された、一又は二以上のスレープ制御機構とを具備し  
て成る制御装置。

## 【請求項2】

マスター制御機構が一又は二以上のプロセッサ上で実行されるマスター制御プロセスであ  
り、且つ一又は二以上のスレープ制御機構が一又は二以上のプロセッサ上で実行される一  
又は二以上のスレーププロセスである、請求項1に記載の制御装置。

## 【請求項3】

マスター制御機構が一又は二以上のマスタープロセッサであり、且つ一又は二以上のスレ 20  
ープ制御機構が一又は二以上のスレーププロセッサである、請求項1に記載の制御装置。

## 【請求項4】

マスター制御機構が、スレープ制御プロセス機構のローディングに基づいて、一又は二以 20  
上のスレープ制御機構間で、処理リソースのサブセットからの一又は二以上の処理リソ  
ースおよび記憶リソースのサブセットからの一又は二以上の記憶リソースの制御を、動的に  
再割り当てするように構成されている、請求項1に記載の制御装置。

## 【請求項5】

マスター制御機構が、スレープ制御プロセス機構のローディングに基づいて、一又は二以  
上の追加のスレープ制御機構を動的に割り当て、且つ処理リソースのサブセットからの一  
又は二以上の処理リソースおよび記憶リソースのサブセットからの一又は二以上の記憶リ  
ソースの制御を、追加された一又は二以上のスレープ制御機構に割り当てるよう構成され 30  
ている、請求項1に記載の制御装置。

## 【請求項6】

マスター制御機構が、スレープ制御プロセス機構のローディングに基づいて、一又は二以  
上のスレープ制御機構が一又は二以上の特定のスレープ制御機構にすでに割り当てられ  
ている、処理リソースのサブセットからの一又は二以上の特定の処理リソースおよび記憶  
リソースのサブセットからの一又は二以上の特定の記憶リソースの制御を、一又は二以  
上のスレープ制御機構からの一又は二以上の他のスレープ制御機構に再割り当てし、且つ  
一又は二以上の特定のスレープ制御機構の割り当て解除を動的に行なうよう構成されてい  
る、請求項1に記載の制御装置。

## 【請求項7】

マスター制御機構が、 40  
一又は二以上のスレープ制御機構の状態を決定し、  
一又は二以上のスレープ制御機構からの一又は二以上の特定のスレープ制御機構が正しく  
応答しない、または機能していない場合に、一又は二以上の特定のスレープ制御機構の再  
起動を試み、且つ  
一又は二以上の特定のスレープ制御機構が再起動できない場合に、  
一又は二以上の新たなスレープ制御機構を開始し、且つ  
一又は二以上の特定のスレープ制御機構から、一又は二以上の新たなスレープ制御機構へ  
処理リソースおよび記憶リソースの制御を再割り当てするように構成されている、請求項  
1に記載の制御装置。

## 【請求項8】

一又は二以上のスレーブ制御機構が、マスター制御機構の状態を決定し、且つマスター制御機構が異常終了したり、もはや適切に機能していない場合に、一又は二以上のスレーブ制御機構から新たなマスター制御機構を選択するように構成されている、請求項1に記載の制御装置。

【請求項9】

マスター制御機構からの一又は二以上の命令が、第1論理リソースグループの予想された処理および記憶必要条件に基づいて生成される、請求項1に記載の制御装置。

【請求項10】

一又は二以上のスレーブ制御機構が更に、マスター制御機構からの一又は二以上の命令に応じて、

処理リソースの第1サブセットの処理リソース数の動的変更と、

記憶リソースの第1サブセットの記憶リソース数の動的変更と、

処理リソースの第1サブセットの処理リソース数および記憶リソースの第1サブセットの記憶リソース数の変更を反映させるための、処理リソースの第1サブセットと記憶リソースの第1サブセットとの間の通信接続の動的変更を行なうよう構成されている、請求項1に記載の制御装置。

【請求項11】

処理リソースの第1サブセットの処理リソース数および記憶リソースの第1サブセットの記憶リソース数の変更が、処理リソースの第1サブセットおよび記憶リソースの第1サブセットの実際のローディングに基づいて、マスター制御機構によって指示される、請求項10に記載の制御装置。

【請求項12】

一又は二以上のスレーブ制御機構が更に、マスター制御機構からの一又は二以上の命令に応じて、処理リソースの第2サブセットおよび記憶リソースの第2サブセットを含む第2論理リソースグループを確立するように構成されており、第2論理リソースグループは、処理リソースの集合から処理リソースの第2サブセットを選択し、処理リソースの集合から記憶リソースの第2サブセットを選択し、且つ処理リソースの第2サブセットを記憶リソースの第2サブセットに通信接続させることにより、第1論理リソースグループから通信分離される、請求項1に記載の制御装置。

【請求項13】

処理リソースの第1サブセットが、一又は二以上の記憶領域ネットワーク（SAN）スイッチを使用することにより、記憶リソースの第1サブセットに通信接続され、処理リソースの第2サブセットが、一又は二以上のSANスイッチを使用することにより、記憶リソースの第2サブセットに通信接続され、第2論理リソースグループが、タギングおよびSANゾーニングを使用することにより、第1論理リソースグループから通信分離される、請求項12に記載の制御装置。

【請求項14】

SANゾーニングが、ポートレベルSANゾーニングまたはLUNレベルSANゾーニングを使用することにより実行される、請求項13に記載の制御装置。

【請求項15】

マスター制御機構が中央制御機構に通信接続され、マスター制御機構が、第1論理リソースグループへのローディング情報を中央制御機構に与えるよう構成され、且つ

マスター制御機構が、中央制御機構から受信した一又は二以上の中央制御命令に基づいて、一又は二以上のスレーブ制御機構への一又は二以上の命令を生成するように構成されている、請求項1に記載の制御装置。

【請求項16】

マスター制御機構を開始する工程と、

マスター制御機構に通信接続され、且つマスター制御機構からの一又は二以上の命令に応

じて、

処理リソースの集合から処理リソースの第1サブセットを選択し、

記憶リソースの集合から記憶リソースの第1サブセットを選択し、且つ

処理リソースの第1サブセットを記憶リソースの第1サブセットに通信接続させることにより、

処理リソースの第1サブセットおよび記憶リソースの第1サブセットを含む第1論理リソースグループを確立するように構成された一又は二以上のスレープ制御機構を開始する工程とを具備して成る処理リソースを管理する方法。

【請求項17】

マスター制御機構を開始する工程が、一又は二以上のプロセッサ上で実行されるマスター制御プロセスを開始する工程を含み、且つ

一又は二以上のスレープ制御機構を開始する工程が、一又は二以上のプロセッサ上で実行される一又は二以上のスレーププロセスを開始する工程を含む、請求項16に記載の方法

【請求項18】

マスター制御機構を開始する工程が、一又は二以上のマスター制御プロセッサを開始する工程を含み、且つ

一又は二以上のスレープ制御機構を開始する工程が、一又は二以上のスレーププロセッサを開始する工程を含む、請求項16に記載の方法。

【請求項19】

スレープ制御プロセス機構のローディングに基づいて、処理リソースのサブセットからの一又は二以上の処理リソースおよび記憶リソースのサブセットからの一又は二以上の記憶リソースの制御を、一又は二以上のスレープ制御機構の間で、動的に再割り当てするマスター制御機構を更に含む、請求項16に記載の方法。

【請求項20】

スレープ制御プロセス機構のローディングに基づいて、一又は二以上の追加のスレープ制御機構を動的に割り当て、且つ処理リソースのサブセットからの一又は二以上の処理リソースおよび記憶リソースのサブセットからの一又は二以上の記憶リソースの制御を、追加された一又は二以上のスレープ制御機構に割り当てるマスター制御機構を更に含む、請求項16に記載の方法。

【請求項21】

スレープ制御プロセス機構のローディングに基づいて、一又は二以上のスレープ制御機構からの一又は二以上の特定のスレープ制御機構にすでに割り当てられている、処理リソースのサブセットからの一又は二以上の特定の処理リソースおよび記憶リソースのサブセットからの一又は二以上の特定の記憶リソースの制御を、一又は二以上のスレープ制御機構からの一又は二以上の他のスレープ制御機構に再割り当てするマスター制御機構を更に含む、請求項16に記載の方法。

【請求項22】

一又は二以上のスレープ制御機構の状態を決定し、一又は二以上のスレープ制御機構からの一又は二以上の特定のスレープ制御機構が応答しないあるいは適切に機能していない場合に、一又は二以上の特定のスレープ制御機構を再起動させるよう試み、且つ

一又は二以上の特定のスレープ制御機構が再起動できない場合に、一又は二以上の新たな制御機構を開始し、且つ一又は二以上の特定のスレープ制御機構から一又は二以上の新たなスレープ制御機構に、処理リソースおよび記憶リソースの制御を再割り当てするマスター制御機構を更に含む、請求項16に記載の方法。

【請求項23】

マスター制御機構の状態を決定し、且つ

マスター制御機構が異常終了したり、もはや適切に機能していない場合に、一又は二以上のスレープ制御機構から新たなマスター制御機構を選択する一又は二以上のスレープ制御

10

20

30

40

50

機構を更に含む、請求項 16 に記載の方法。

【請求項 24】

マスター制御機構からの一又は二以上の命令が、第 1 論理リソースグループの予測された処理および記憶必要条件に基づいて生成される、請求項 16 に記載の方法。

【請求項 25】

マスター制御機構からの一又は二以上の命令に応じて、  
処理リソースの第 1 サブセットにおける処理リソースの数の動的変更と、  
記憶リソースの第 1 サブセットにおける記憶リソースの数の動的変更と、  
処理リソースの第 1 サブセットにおける処理リソースの数と記憶リソースの第 1 サブセットにおける記憶リソースの数の変更を反映させるための、処理リソースの第 1 サブセットおよび記憶リソースの第 1 サブセットとの間の通信接続の動的変更を行なう一又は二以上のスレープ制御機構を更に含む、請求項 16 に記載の方法。 10

【請求項 26】

処理リソースの第 1 サブセットにおける処理リソースの数および記憶リソースの第 1 サブセットにおける記憶リソースの数の変更が、処理リソースの第 1 サブセットおよび記憶リソースの第 1 サブセットの実際のローディングに基づいて、マスター制御機構によって指示される、請求項 25 に記載の方法。

【請求項 27】

マスター制御機構からの一又は二以上の命令に応じて、処理リソースの第 2 サブセットおよび記憶リソースの第 2 サブセットを含む第 2 論理リソースグループを確立する一又は二以上のスレープ制御機構を更に含み、第 2 論理リソースグループが、  
処理リソースの集合からの処理リソースの第 2 サブセットの選択、  
処理リソースの集合からの記憶リソースの第 2 サブセットの選択、且つ  
記憶リソースの第 2 サブセットへの処理リソースの第 2 サブセットの通信接続によって、  
第 1 論理リソースグループから情報伝達分離されている、請求項 16 に記載の方法。 20

【請求項 28】

処理リソースの第 1 サブセットが、一又は二以上の記憶領域ネットワーク（SAN）スイッチを使用することによって記憶リソースの第 1 サブセットに通信接続され、  
処理リソースの第 2 サブセットが、一又は二以上の SAN スイッチを使用することによって記憶リソースの第 2 サブセットに通信接続され、且つ  
第 2 論理リソースグループが、タギングおよび SAN ゾーニングすることによって第 1 論理リソースグループから通信分離されている、請求項 27 に記載の方法。 30

【請求項 29】

SAN ゾーニングが、ポートレベル SAN ゾーニングまたは LUN レベル SAN ゾーニングを使用することにより実行される、請求項 28 に記載の方法。

【請求項 30】

マスター制御機構が中央制御機構に通信接続され、  
マスター制御機構が、第 1 論理リソースグループのローディング情報を中央制御機構に与えるよう構成され、  
マスター制御機構がさらに、中央制御機構から受信した一又は二以上の中央制御命令に基づいて、一又は二以上のスレープ制御機構の一又は二以上の命令を生成するように構成されている、請求項 16 に記載の方法。 40

【請求項 31】

処理リソースを管理するための一又は二以上の命令の一又は二以上のシーケンスを伝えるコンピュータで読み取り可能な媒体であって、一又は二以上のプロセッサで一又は二以上の命令の一又は二以上のシーケンスを実行すると、一又は二以上のプロセッサが、  
マスター制御機構を開始させる工程と、  
マスター制御機構に通信接続され、且つマスター制御機構からの一又は二以上の命令に応じて、処理リソースの第 1 サブセットおよび記憶リソースの第 1 サブセットを含む第 1 論理リソースグループを確立するように構成されている一又は二以上のスレープ制御機構を 50

開始させる工程とを、  
処理リソースの集合から処理リソースの第1サブセットを選択し、  
記憶リソースの集合から記憶リソースの第1サブセットを選択し、且つ  
記憶リソースの第1サブセットに処理リソースの第1サブセットを通信接続させることによ  
って行なう、コンピュータで読み取り可能な媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は一般に、データ処理に関する。本発明は、特にコンピューティンググリッドを制  
御する方法および装置に関する。

【0002】

【発明が解決しようとする課題】

今日のウェブサイトおよび他のコンピュータシステムのビルダーは、多くの興味深いシス  
テムプランニング問題を抱えている。これらの問題には、容量プランニング、サイト利用  
可能度およびサイトの安全性が含まれている。これらの目標を達成するには、潜在的に大  
きく複雑であるかもしれないサイトの設計および運営が可能なトレーニングを受けた人員  
を探し出して、雇用することが必要である。多くの組織にとって、大きなサイトの設計、  
構築、運営は主力事業でないことが多いため、このような人員を探し出して雇用すること  
は難しいことが分かっている。

【0003】

1つの方法として、他の企業の他のウェブサイト共に同じ場所に配置された、第三者サイ  
トの企業ウェブサイトを採用した。このような外部委託施設は現在、Exodus、Ab  
oveNet、GlobalCenterなどの企業から利用できる。これらの施設によ  
り、多数の顧客が共有する物理的スペース、冗長ネットワーク、発電施設が与えられる。

【0004】

外部委託ウェブサイトの採用により、ウェブサイトの確立と維持の負担が大きく減るが、  
企業からウェブサイトの維持に関連する全ての問題を取り除くことにはならない。企業は  
、その施設の構築、運営、増大の間に、そのコンピューティング構造基盤に関する多くの  
仕事を行わなければならない。このような施設で採用された企業の情報テクノロジー管  
理者は、施設でのその演算装置の手動選択、設置、構成、維持に関して責任がある。管理  
者は、リソースプランニングおよび取り扱いピーク容量などの困難な問題に取り組まな  
なければならない。特に、管理者は、需要に対処するために外部委託企業からリソース需要お  
よび要求リソースを予測する必要がある。多くの管理者は、予期しないピーク需要に対す  
る緩和策として、必要とする以上に実質的に多いリソースを要求することで十分な容量を  
確保する。残念ながら、これによって未使用の容量が多大なものになり、ウェブサイトを  
採用するための企業の諸経費が増加してしまう。

【0005】

外部委託企業も、サーバ、ソフトウェア、電力施設を含む完全計算施設を提供しても、成  
長に伴って同一の手動の誤りやすい管理処置が必要となるので、外部委託企業にとって施  
設の拡大および成長は簡単ではない。さらに、予期しないピーク需要に対する容量プラン  
ニングと共に問題が残っている。この場合、外部委託企業は、かなりの量の未使用容量を  
維持することができる。

【0006】

さらに、外部委託企業が管理するウェブサイトの必要条件は異なることがしばしばある。  
例えば、ある企業では、そのウェブサイトを独立して運営および制御するための能力が必  
要になる。他の企業では、そのウェブサイトを、外部委託企業と共に配置された他の全て  
のサイトから分離させる特定の種類またはレベルの安全保障が必要となる。別の例として  
、ある企業では、どこかに配置された企業イントラネットへの確実な接続が必要となる。

【0007】

さらに、様々なウェブサイトは、内部トポロジにおいて異なる。あるサイトは単に、ウ

ウェブロードバランスによってロードバランスの取れたウェブサーバ列から構成される。適切なロードバランスはCisco Systems, Inc.のLocal Director、F5 LabsのBigIP、AlettonのWeb Directorなどである。他のサイトは多層構成されることもあり、これによってウェブサーバ列はハイパーテキストプロトコル(HTTP)要求に対処できるが、アプリケーションロジックの大半は別のアプリケーションサーバにおいて実施される。これらのアプリケーションサーバを、データベースサーバの層に再び接続しなければならないことがある。

【0008】

このような異なる構造シナリオの幾つかを、図1A、図1B、図1Cに示す。図1Aは単純なウェブサイトのブロック図であり、CPU102およびディスク104を含む単一のコンピュータ要素またはマシン100から成る。マシン100は、インターネットとして知られる世界規模のパケット交換式データネットワーク106、または他のネットワークに接続されている。マシン100は、上述したタイプの同一位置サーバ内に収容されていてもよい。

10

【0009】

図1Bは、複数のウェブサーバWSA、WSB、WSCを含む1層ウェブサーバファーム110のブロック図である。各ウェブサーバは、インターネット106に接続されたロードバランス112に接続されている。ロードバランスはサーバ間のトラフィックを分割して、各サーバのバランスのとれた処理ロードを維持する。ロードバランス112も、ウェブサーバを許可されていないトラフィックから保護するためのファイアウォールを含むか、あるいはこれに接続されていてもよい。

20

【0010】

図1Cは、ウェブサーバW1、W2などの層、アプリケーションサーバA1、A2などの層、およびデータベースサーバD1、D2などの層を含む3層サーバファーム120を示す。ウェブサーバは、HTTP要求に対処するために設けられる。アプリケーションサーバは、アプリケーションロジックの大部分を実行する。データベースサーバは、データベース管理システム(DBMS)ソフトウェアを実行する。

【0011】

構成する必要があるウェブサイトの種類のトポロジーが多様化され、該当する企業の必要条件が変化しているので、大規模ウェブサイトを構成する唯一の方法は、各サイトを物理的にカスタマイズすることであると考えられる。多くの組織はそれぞれ個別に同一問題に取り組んでおり、ゼロから各ウェブサイトをカスタマイズしている。これは非能率的であり、異なる企業で大量の同一の仕事が生じることになる。

30

【0012】

従来の方の別の問題は、リソースと容量プランニングである。ウェブサイトは、異なる日、またはその日の異なる時間で、非常に異なるレベルのトラフィックを受信する。ピークトラフィック時間では、ウェブサイトのハードウェアまたはソフトウェアは、オーバーロードのために適当な時間内で要求に応答することができないことがある。他の時間では、ウェブサイトのハードウェアまたはソフトウェアには過度の容量があり、十分に利用されていない。従来の方では、過度のコストを負ったり過剰容量となることなく、ピークトラフィックに対処する十分なハードウェアおよびソフトウェアを有することにおけるバランスを見つけることは、困難な問題である。多くのウェブサイトは適切なバランスを見つけることができず、慢性的に過小容量または過剰容量に悩まされている。

40

【0013】

別の問題は、ヒューマンエラーによって引き起こされる故障である。手動構成されたサーバファームを使用する現在の方法において存在する大きな潜在的災害は、新しいサーバをライブサーバ内に構成するときのヒューマンエラーにより、サーバファームが誤動作し、これによってウェブサイトのユーザへのサービスが失われてしまう可能性があることである。

【0014】

50

上記に基づき、この分野において、カスタム構成を必要とすることなく、要求があり次第、直ちに簡単に拡張することのできるコンピューティングシステムを提供する改善された方法および装置が明確に必要である。

【0015】

さらに、トラフィックスループットの変化を明らかにするためにそれぞれ必要に応じて拡張または縮小可能な多数の分離処理ノードの生成をサポートするコンピューティングシステムも必要である。

【0016】

さらに、このような拡張可能コンピューティングシステムとその構成分離処理ノードを制御する方法および装置も必要である。他の必要性もここに示す開示内容から明らかとなるであろう。

【0017】

【発明の開示】

本発明の1つの態様によれば、上記必要性、および以下の説明により明らかとなる他の必要性は、大規模なコンピューティング構造（「コンピューティンググリッド」）に基づき、非常に拡張性があり、非常に利用しやすくして確実なデータ処理サイトを制御および管理する方法および装置によって達せられる。コンピューティンググリッドは、物理的に構成され、その後要求に応じて様々な組織に対して論理的に分割される。コンピューティンググリッドは、一又は二以上のVLANスイッチおよび一又は二以上の記憶領域ネットワーク（SAN）スイッチに接続された非常に多数のコンピューティング要素を含んでいる。複数の記憶装置はSANスイッチに接続され、且つ適切な切り替えロジックおよびコマンドを介して、一又は二以上のコンピューティング要素に選択的に接続されてもよい。VLANスイッチの1つのポートは、インターネットなどの外部ネットワークに接続される。監視機構、層、マシンまたはプロセスは、VLANスイッチおよびSANスイッチに接続される。

【0018】

初めに、全ての記憶装置およびコンピューティング要素は、アイドルブールに割り当てられる。プログラム制御の下、監視機構はVLANスイッチおよびSANスイッチのポートを一又は二以上のコンピューティング要素および記憶装置に接続するように動的に構成する。その結果、このような要素および装置はアイドルブールから論理的に除去されて、一又は二以上の仮想サーバファーム（VSF）またはインスタントデータセンタ（IDC）の一部となる。各VSFコンピューティング要素は、ブートストラップ操作および生成実行を行なうためにコンピューティング要素が使用できるブートイメージを含む記憶装置に向けられるか、あるいは関連付けられる。

【0019】

本発明の1つの態様によると、監視層は、一又は二以上のスレーブ制御プロセス機構に通信接続された一又は二以上のマスター制御プロセス機構を含む制御機構階層から成る制御プレーンである。一又は二以上のマスター制御プロセス機構は、スレーブ制御プロセス機構のロードインギングに基づいて、スレーブ制御プロセス機構を割り当ておよび割り当て解除する。一又は二以上のマスター制御プロセス機構は、処理および記憶リソースのサブセットを選択することによってIDCを確立するようにスレーブ制御プロセス機構に支持する。一又は二以上のマスター制御プロセス機構は、スレーブ制御プロセス機構の周期的な検診を行なう。応答がなかったり、あるいは異常終了したスレーブ制御機構は再起動される。別のスレーブ制御機構は開始されて、再開できないスレーブ制御機構の代わりとなる。スレーブ制御機構は、マスター制御機構の周期的な検診を行なう。マスタースレーブ制御プロセス機構が異常終了すると、スレーブ制御プロセス機構が選択されて新たなマスター制御プロセス機構となり、以上終了したマスター制御プロセス機構の代わりとなる。

【0020】

コンピューティンググリッドを一度物理的に構成し、且つ要求に応じてコンピューティンググリッドの部分の部分を確実且つ動的に様々な組織に割り当てることにより、各サイトのカス

タマイズのときには困難であったスケールメリットが得られる。

【0021】

本発明は、添付の図面において、限定するのではなく、一例として図解されており、且つその中において同一の参照番号は同様の要素を示している。

【0022】

【本発明の実施の形態】

以下の説明において、説明の目的で、本発明を完全に理解してもらうために多数の特定の細部が述べられている。しかしながら、本発明がこれらの特定の細部無しに実施されることは当業者に明らかとなるであろう。他の例では、本発明が不必要に分かりにくくしてしまうのを回避するために、既知の構造および装置がブロック図で示されている。

10

【0023】

仮想サーバファーム (VSF)

一実施例によると、大規模なコンピューティング構造（「コンピューティンググリッド」）が設けられる。コンピューティンググリッドは物理的に一度構成され、その後要求に応じて論理的に区画されてもよい。コンピューティンググリッドの一部は、複数の企業または組織のそれぞれに割り当てられる。各組織のコンピューティンググリッドのロジック部分には、仮想サーバファーム (VSF) と呼ばれる。各組織はその VSF の独立した運営管理制御を維持する。各 VSF は、サーバファームまたは他の要素に与えられたリアルタイム要求に基づいて、CPU の数、記憶容量およびディスク、ネットワーク帯域幅を動的に変更することができる。VSF は同一の物理的コンピューティンググリッドから全て論理的に生成されるが、各 VSF は全てのほかの組織の VSF から保護されている。VSF は、イントラネットを他の組織の VSF にさらすことなく、個人専用回線または仮想プライベートネットワーク (VPN) を使用することにより、イントラネットに逆に接続することができる。

20

【0024】

組織は、コンピュータへの完全（例えば、スーパーユーザまたはルート）管理アクセスを実行し、これらのコンピュータが接続されたローカルエリアネットワーク (LAN) の全てのトラフィックを観察することができるが、それに割り当てられたコンピューティンググリッドの部分、つまり VSF におけるデータおよびコンピューティング要素のみアクセスできる。一実施例によると、これは、VSF の安全限界が動的に拡張および縮小する動的ファイアウォール方式を使用することによって可能となる。各 VSF を使用して、インターネット、イントラネットまたはエキストラネットを介してアクセスできる組織の内容とアプリケーションを採用することができる。

30

【0025】

コンピューティング要素およびその関連するネットワークキング、および記憶要素の構成と制御は、コンピューティンググリッドにおけるコンピューティング要素の何れかによって直接アクセスすることのできない監視機構によって行なわれる。便宜上、本文書では、監視機構は一般に制御プレーンと呼ばれ、一又は二以上のプロセッサまたはプロセッサのネットワークから構成されていてもよい。監視機構は、スバパイザ、コントローラなどで構成されていてもよい。ここに説明するように、他の方法を用いることもできる。

40

【0026】

制御プレーンは、例えばネットワーク内または他の手段によって相互接続される一又は二以上のサーバなど、監視の目的用に割り当てられたコンピューティング要素の完全独立集合上で実施される。制御プレーンは、グリッドのネットワークキングおよび記憶要素の特殊制御ポートまたはインタフェースを介して、コンピューティンググリッドのコンピューティング、ネットワークキングおよび記憶要素に対して、制御動作を行なう。制御プレーンはシステムの切り替え要素に物理的インタフェースを与え、システムにおけるコンピューティング要素の負荷を監視し、グラフィカルユーザインタフェースまたは他の適切なユーザインタフェースを使用して運営管理機能を与える。

【0027】

50

制御プレーンを実施するのに使用するコンピュータはコンピュータグリッド（および特定の VSF）におけるコンピュータには論理的には不可視であり、コンピュータグリッドにおける要素を介して、あるいは外部コンピュータから、決して攻撃されたり、破壊されることはない。制御プレーンのみがコンピュータグリッドにおける機器の制御ポートへの物理的接続部を有しており、これは特定の VSF におけるメンバーシップを制御する。コンピュータリングにおける機器はこれらの特殊制御ポートを介してのみ構成できるので、コンピュータリンググリッドにおけるコンピュータリング要素はその安全限界を変更したり、認められていない記憶またはコンピュータリング機器へのアクセスを行なうことはできない。

#### 【0028】

従って、VSF により、組織は、大規模共有コンピュータリングインフラストラクチャから動的に作られたプライベートサーバファーム、すなわちコンピュータリンググリッドから構成されたように見えるコンピュータリング設備と連動することができる。ここに説明するコンピュータリングアーキテクチャと接続された制御プレーンは、そのプライバシーと安全性がコンピュータリンググリッドの機器のハードウェアにおいて実施されるアクセス制御機構によって保護されるプライベートサーバファームを与える。

#### 【0029】

制御プレーンは、各 VSF の内部トポロジーを制御する。制御プレーンはここに説明するコンピュータ、ネットワークスイッチおよび記憶ネットワークスイッチの基本相互接続を取り、これらを使用して様々なサーバファーム構成を作成することが可能である。これらには、限定されるものではないが、ロードバランサによって前処理された単層ウェブサーバファーム、および多層構成が含まれており、ウェブサーバはアプリケーションサーバと通信し、且つアプリケーションサーバはデータベースサーバと通信を行なう。様々な負荷バランシング、多層化、ファイアウォール構成が可能である。

#### 【0030】

コンピュータリンググリッド

コンピュータリンググリッドは単一の場所に存在し、幅広い領域に分散させることができる。最初に、本書はローカルエリア技術でのみ構成される単一の建物のサイズのネットワークにおける、コンピュータグリッドについて説明する。次に、本書は、コンピュータグリッドを広域ネットワーク（WAN）上で分散させる場合について説明する。

#### 【0031】

図2は、ローカルコンピュータリンググリッド208を含む拡張可能コンピュータリングシステム200の1つの構成を示すブロック図である。本書において、「拡張可能」とは一般に、システムがフレキシブルで拡張性があり、要求があり次第特定の企業またはユーザに対して低下あるいは上昇させた計算力を与える能力を有していることを意味する。ローカルコンピュータリンググリッド208は、多数のコンピュータリング要素 CPU1、CPU2、...、CPU<sub>n</sub> から成る。実施例において、10、000個以上のコンピュータリング要素が存在している。これらのコンピュータリング要素は長期の要素ごとの状態情報を含んでいたり、保存することはないので、ローカルディスクなどの永続性または不揮発性ストレージなしで構成してもよい。その代わり、全ての長期の状態情報は、コンピュータリング要素とは別に、一又は二以上の SAN スイッチ 202 を含む記憶領域ネットワーク（SAN）を介してコンピュータリング要素に接続される複数のディスク、ディスク1、ディスク2、...、ディスク<sub>n</sub> に保存される。適切な SAN スwitch の例は、Brocade および Exce1 から販売されている。

#### 【0032】

全てのコンピュータリング要素は、仮想 LAN（VLAN）に分割される一又は二以上の VLAN スイッチ 204 を介して、相互接続される。VLAN スイッチ 204 はインターネット 106 に接続されている。一般に、コンピュータリング要素は、VLAN スイッチに接続された1つまたは2つのネットワークインタフェースを含んでいる。便宜上、図2において、全てのノードが2つのネットワークインタフェースを有しているが、ネットワ

ークインタフェースがこれよりも少ないまたは多いノードもある。多くの製造供給元は現在、VLAN機能をサポートするスイッチを提供している。例えば、適切なVLANスイッチはCisco Systems, IncおよびXtreme Networksより入手可能である。同様に、SANを構成するための入手可能製品は多数あり、これにはファイバーチャネルスイッチ、SCSI対ファイバーチャネルブリッジング機器、ネットワークアタッチドストレージ(NAS)機器が含まれる。

#### 【0033】

制御プレーン206は、SAN制御経路、CPU制御経路、およびVLAN制御経路によって、SANスイッチ202、CPU1、CPU2、... CPU<sub>n</sub>およびVLANスイッチ204にそれぞれ接続される。

#### 【0034】

各VSFは、VLANの集合、VLANに取り付けられるコンピューティング要素の集合、およびコンピューティング要素の集合に接続されるSAN上で利用可能な記憶装置のサブセットから成る。SAN上で利用可能なストレージのサブセットをSANゾーンと呼び、これはSANハードウェアによって他のSANゾーンの一部であるコンピューティング要素からのアクセスから保護されている。好適には、非可鍛性ポート識別子を与えるVLANを使用して、一人の顧客またはエンドユーザが他の顧客またはエンドユーザのVSFリソースにアクセスするのを防止する。

#### 【0035】

図3は、SANゾーンを特色とする典型的な仮想サーバファームのブロック図である。複数のウェブサーバWS1、WS2などは、第1VLAN(VLAN1)によってロードバランサ(LB)/ファイアウォール302に接続されている。第2VLAN(VLAN2)は、インターネット106をロードバランサ(LB)/ファイアウォール302に接続する。各ウェブサーバは、後に説明する機構を使用してCPU1、CPU2などから選択することができる。ウェブサーバはSANゾーン304に接続されており、これは一又は二以上の記憶装置306a、306bに接続されている。

#### 【0036】

ある時点において、例えば図2のCPU1などのコンピューティンググリッドにおけるコンピューティング要素は、VLANの集合および単一のVSFに関連するSANゾーンに接続されているだけである。通常、VSFは異なる組織間で共有されることはない。単一のSANゾーンに属するSAN上のストレージのサブセット、およびそれに関連するVLANの集合、およびこれらのVLAN上のコンピューティング要素が、VSFを規定する。

#### 【0037】

VLANのメンバーシップおよびSANゾーンのメンバーシップを制御することにより、制御プレーンはコンピューティンググリッドを多数のVSFに論理分割する。1つのVSFのメンバーは、他のVSFのコンピューティングまたは記憶リソースにアクセスできない。このようなアクセス制限は、VLANスイッチによって、且つファイバーチャネルスイッチとSCSI対ファイバーチャネルブリッジングハードウェアなどのエッジ機器といったSANハードウェアのポートレベルアクセス制御機構(例えばゾーニング)によって実行させる。コンピューティンググリッドの一部を形成するコンピューティング要素はVLANスイッチおよびSANスイッチの制御ポートまたはインタフェースに物理的に接続されていないので、VLANまたはSANゾーンのメンバーシップを制御することはできない。従って、コンピューティンググリッドのコンピューティング要素は、これらを含むVSFに配置されていないコンピューティング要素にアクセスできない。

#### 【0038】

制御プレーンを実行するコンピューティング要素のみが、グリッドにおける機器の制御ポートまたはインタフェースに物理的に接続される。コンピューティンググリッドの機器(コンピュータ、SANスイッチ、およびVLANスイッチ)は、これらの制御ポートまたはインタフェースによって構成されるだけである。これにより、コンピューティンググリ

ッドを多数のV S Fに動的に分割する単純であるが非常に安定した手段が得られる。

#### 【0039】

V S Fにおける各コンピューティング要素は、他のコンピューティング要素と交換可能である。あるV S Fに関連するコンピューティング要素、V L A NおよびS A Nゾーンの数は、制御プレーンの制御の下で時間が経つと変化する。

#### 【0040】

一実施例において、コンピューティンググリッドは、予備の多数のコンピューティング要素から成るアイドルプールを含んでいる。アイドルプールからのコンピューティング要素は、C P Uの増加、そのV S Fで利用可能なメモリ容量、あるいはV S Fにおける特定のコンピューティング要素の故障に対する対処などの理由で、特定のV S Fに割り当ててもよい。コンピューティング要素がウェブサーバとして構成されている場合、アイドルプールは、変化するあるいは「バースト状の」ウェブトラフィック負荷および関連するピーク処理負荷に対する大きな「ショックアブソーバ」として機能する。

#### 【0041】

アイドルプールは多数の異なる組織間で共有されるので、単一の組織がアイドルプール全体の費用を支払わなければならないということがないため、スケールメリットが得られる。異なる組織が必要に応じてその日の異なる時間でアイドルプールからコンピューティング要素を得ることができるので、各V S Fは必要ときに拡大し、かつトラフィックが通常の状態に落ち着いたときに縮小することが可能となる。多数の異なる組織が同時にピークに達し続け、それによってアイドルプールの容量が使い果たされる可能性がある場合、アイドルプールはそれに更に多くのC P Uと記憶要素を追加することで増大させることが可能である(拡張性)。アイドルプールの容量は、通常の状態において、特定のV S Fが必要ときにアイドルプールから別のコンピューティング要素を得ることができない確率を大きく減らすよう設計されている。

#### 【0042】

図4 A、図4 B、図4 Cおよび図4 Dは、アイドルプールからコンピューティング要素を出し入れするときの連続工程を示すブロック図である。最初に図4 Aを参照し、制御プレーンがコンピューティンググリッドの要素を、V S F 1およびV S F 2というラベルの第1および第2V S Fに論理的に接続させたものとする。アイドルプール400は複数のC P U 402から成り、そのうちの1つはC P U Xとラベル付けされている。図4 Bにおいて、V S F 1で別のコンピューティング要素が必要となった。従って、制御プレーンは、経路404で示すように、C P U Xをアイドルプール400からV S F 1に移動させる。

#### 【0043】

図4 Cにおいて、V S F 1はもはやC P U Xが必要ではないので、制御プレーンはC P U XをV S F 1からアイドルプール400に戻す。図4 Dにおいて、V S F 2で別のコンピューティング要素が必要となった。従って、制御プレーンはC P U Xをアイドルプール400からV S F 2に移動させる。従って、時間が経過して、トラフィックの状態が変化すると、単一のコンピューティング要素がアイドルプールに属し(図4)、特定のV S Fに割り当てられ(図4 B)、アイドルプールに戻され(図4 C)、そして別のV S Fに属することとなる(図4 D)。

#### 【0044】

これらの各段階において、制御プレーンは、特定のV S F(またはアイドルプール)に関連するV L A NおよびS A Nゾーンの一部となるそのコンピューティング要素に関連するL A NスイッチおよびS A Nスイッチを構成する。一実施例によると、各推移の間において、コンピューティング要素はパワーダウンまたは再起動される。コンピューティング要素の電源が再び投入されると、コンピューティング要素はS A Nの記憶ゾーンの異なる部分を見る。特に、コンピューティング要素は、オペレーティングシステム(例えば、L i n u x、N T、S o l a r i sなど)の起動可能イメージを含むS A N上の記憶ゾーンの部分を見る。記憶ゾーンはまた各組織に特有のデータ部分を含む(例えば、ウェブサーバに関連するファイル、データベースパーティションなど)。コンピューティング要素はま

た別のV S FのV L A N集合の一部である別のV L A Nの一部であるため、転送先のV S FのV L A Nに関連するC P U、S A N記憶装置、N A S機器にアクセスできる。

#### 【0045】

好適な実施例において、記憶ゾーンは、コンピューティング要素によって想定される役割に関連する複数の予め定義された論理詳細設計を含んでいる。初めに、何れのコンピューティング要素も、ウェブサーバ、アプリケーションサーバ、データベースサーバなどの特定の役割やタスクにあてがわれていない。コンピューティング要素の役割は複数の予め定義された保存された詳細設計の何れかから得られ、このような詳細設計のそれぞれはその役割に関連するコンピューティング要素のブートイメージを定義する。詳細設計は、ブートイメージ位置を役割に関連付けさせるファイル、データベーステーブル、または他の保存形式で保存される。

#### 【0046】

従って、図4A、図4B、図4Cおよび図4DにおけるC P U Xの移動は論理的であって、物理的ではなく、制御プレーンの制御の下でV L A NスイッチおよびS A Nゾーンを再構成することによって行なわれる。また、コンピューティンググリッドにおける各コンピューティング要素はまず本来代替可能であり、仮想サーバファームに接続されてブートイメージからソフトウェアをロードした後でのみ特定の処理役割を想定する。何れのコンピューティング要素も、ウェブサーバ、アプリケーションサーバ、データベースサーバなどの特定の役割またはタスクがあてがわれていない。コンピューティング要素の役割は、複数の予め定義された保存された詳細設計の何れかから得られ、これらの詳細設計のそれぞれは役割に関連しており、役割に関連するコンピューティング要素のブートイメージを定義する。

#### 【0047】

長期の状態情報は特定のコンピューティング要素（ローカルディスクなど）に保存されていないので、異なるV S F間でノードは簡単に移動でき、まったく異なるO Sおよびアプリケーションソフトウェアを実行させることができる。これにより、計画された、あるいは計画されていないダウンタイムの場合に、コンピューティング要素はより交換しやすくなる。

#### 【0048】

特定のコンピューティング要素は、様々なV S Fから出し入れするときに、異なる役割を実行することができる。例えば、コンピューティング要素は、あるV S Fにおいてウェブサーバとして動作し、且つ別のV S Fに移動させると、データベースサーバ、ウェブロードバランサ、ファイアウォールなどになる。また、異なるV S Fにおいて、Linux、NT、Solarisなどの異なるオペレーティングシステムを連続的に起動および実行することもできる。従って、コンピューティンググリッドにおける各コンピューティング要素は代替可能であり、それに割り当てられた固定的役割はない。従って、コンピューティンググリッドの予備容量全体を使用して、何れかのV S Fが必要とする何らかのサービスを提供することができる。これにより、特定のサービスを実行する各サーバが有する同一のサービスを提供することのできるバックアップサーバの数は数千になるので、単一のV S Fが提供するサービスの利用可能度および信頼性は非常に高くなる。

#### 【0049】

さらに、コンピューティンググリッドの高予備容量によって、動的負荷バランシング特性および高プロセッサ利用可能度が得られる。この能力は、V L A Nを介して相互接続され、S A Nを介して記憶装置の構成可能ゾーンに接続され、また全て制御プレーンによってリアルタイムで制御されるディスクレスコンピューティング要素の一義的な組合せで可能となる。各コンピューティング要素はV S Fにおける何れかの必要サーバの役割において動作することができ、またS A Nにおける何れかのディスクの何れかの論理分割に接続可能である。グリッドで更なるコンピューティングパワーやディスク容量が必要な場合、コンピューティング要素またはディスクストレージはアイドルプールに手動で追加されるが、これは時間が経過して更に多くの組織にV S Fサービスが提供されると減少する。C P

Uの数、ネットワークおよびディスク処理能力、VSFで利用できる記憶装置を増大させるのに、手動で介入する必要はない。これらのリソース全ては、要求があるたびにCPU、アイドルプールの状態で利用できるネットワークおよびディスクリソースから、制御プレーンによって割り当てられる。

#### 【0050】

特定のVSFは、手動で再構成されない。アイドルプールのコンピューティング要素のみが、コンピューティンググリッドに手動で再構成される。その結果、現在手動で構成されたサーバファームに存在する大きな潜在的障害が除去される。新たなサーバをライブサーバファームに構成する際のヒューマンエラーによってサーバファームが誤動作し、その結果そのウェブサイトのユーザへのサービスが失われてしまう可能性は、殆どなくなる。

#### 【0051】

制御プレーンはまた、SANに取り付けられた記憶装置に保存されたデータをコピーするので、特定の記憶要素の故障によって、システムの何れかの部分へのサービスが失われることはない。SANを使用し、且つ冗長的な記憶およびコンピューティング要素を与えることで、コンピューティング装置から長期記憶装置を取り除くことにより、どのコンピューティング要素も何れかの記憶パーティションに取り付けることができるので、高い利用可能性が得られる。

#### 【0052】

仮想サーバファームの確立、それに対するプロセッサの追加、およびそれからのプロセッサの除去の詳細な例

図5は、実施例によるコンピューティンググリッドおよび制御プレーン機構のブロック図である。図5を参照し、以下においてVSFを作成し、それにノードを追加し、且つそれからノードを除去するのに使用できる詳細な過程を説明する。

#### 【0053】

図5は、VLANケーパブルスイッチ504に接続されたコンピュータA~Gを含むコンピューティング要素502を示す。VLANスイッチ504はインターネット106に接続されており、且つVLANスイッチはポートV1、V2などを有している。コンピュータA~Gは更にSANスイッチ506に接続され、これは複数の記憶装置またはディスクD1~D5に接続されている。SANスイッチ506はポートS1、S2などを有している。制御プレーン機構508は、制御経路およびデータ経路によって、SANスイッチ506およびVLANスイッチ504に通信接続されている。制御プレーンは、制御ポートを介してこれらの装置に制御コマンドを送信することができる。

#### 【0054】

便宜上、図5のコンピューティング要素の数は少くなっている。実際には、多数のコンピュータ、例えば数千以上、および同数の記憶装置がコンピューティンググリッドを形成している。このような大きな構造において、多数のSANスイッチは相互接続されてメッシュを形成し、且つVLANスイッチは相互接続されてVLANメッシュを形成している。しかしながら、分かりやすくするため、図5では単一のSANスイッチと単一のVLANスイッチを示している。

#### 【0055】

まず、全てのコンピュータA~Gが、制御プレーンがVSFの作成要求を受信するまで、アイドルプールに割り当てられている。VLANスイッチの全てのポートは、(アイドル用)VLAN Iとラベル付けされる特定のVLANに割り当てられている。制御プレーンがVSFを構成するように要求され、SAN上の記憶装置に接続された1つのロードバランサ/ファイアウォールおよび2つのウェブサーバを含むものとする。制御プレーンへの要求は、管理インタフェースまたは他のコンピューティング要素を介して受信される。

#### 【0056】

それに応じて、制御プレーンはCPU Aをロードバランサ/ファイアウォールとして指定または割り当て、且つCPU BおよびCPU Cをウェブサーバとして割り当てる。CPU

10

20

30

40

50

Aは論理的にSANゾーン1に置かれ、専用のロードバランサ/ファイアウォールソフトウェアを含むディスク上の起動可能パーティションに向けられる。「向けられる」という語は便宜上使用され、いかなる手段によって、動作させる必要のある適切なソフトウェアをCPUAが入手あるいは探し出すのに十分な情報がCPUAに与えられることを意味する。SANゾーン1にCPUAを配置することにより、CPUAは、そのSANゾーンのSANによって制御されるディスクからリソースを得ることが可能になる。

【0057】

ロードバランサは、負荷バランスすべき2つのウェブサーバとしてのCPUBおよびCPUCについて知るために、制御プレーンによって構成される。ファイアウォール構成は、インターネット106からの認められないアクセスから、CPUBおよびCPUCを保護する。CPUBおよびCPUCは、特定のオペレーティングシステム（例えば、Solaris、Linux、NTなど）およびウェブサーバアプリケーションソフトウェア（例えばApache）のための起動用OSイメージを含むSAN上のディスクパーティションに向けられる。VLANスイッチは、VLAN1にポートv1およびv2を配置し、且つVLAN2にポートv3、v4、v5、v6およびv7を配置するように構成される。制御プレーンはSANスイッチ506を構成して、ファイバチャネルスイッチポートs1、s2、s3およびs8をSANゾーン1に配置する。

【0058】

CPUがどのように特定のディスクドライブに向けられ、且つこれが起動とディスクデータへの共有アクセスにどのような意味があるのかをここに説明する。

【0059】

図6は、まとめてVSF1と呼ばれるコンピューティング要素の論理接続の結果を示すブロック図である。ディスクドライブDD1は記憶装置D1、D2などから選択される。図6に示す論理構造が得られると、CPUA、B、Cには起動コマンドが与えられる。それに応じて、CPUAは専用ロードバランサ/ファイアウォールコンピューティング要素となり、且つCPUBおよびCPUCはウェブサーバとなる。

【0060】

今、方針に基づく規則のために、制御プレーンが、VSF1において別のウェブサーバが必要であると判断したものとする。これは、例えば、ウェブサーバへの要求の増加によって起こるものであり、且つ顧客の計画によって少なくとも3つのウェブサーバをVSF1に追加することが可能となる。あるいは、VSFを所有または運営する組織が別のサーバを欲し、そのVSFに更にサーバを追加することの可能な特権的ウェブページなどの管理機構によって追加したためである。

【0061】

それに応じて、制御プレーンはVSF1にCPUDを追加することを決定する。そのために、制御プレーンは、ポートv8およびv9をVLAN2に追加することで、VLAN2にCPUDを追加する。また、CPUDのSANポートs4はSANゾーン1に追加される。CPUDは、ウェブサーバとして起動および実行されるSAN記憶装置の起動可能部分に向けられる。CPUDはまたウェブページ内容、実行可能サーバスクリプトなどから成るSAN上の共有データに読み出し専用アクセスする。このように、CPUBおよびCPUCが要求に対応するように、サーバファームに向けられたウェブ要求に対処することができる。制御プレーンは、CPUDを負荷バランシングされているサーバセットの一部として含むようロードバランサ（CPUA）を構成する。

【0062】

次にCPUDは起動され、VSFのサイズは3つのウェブサーバおよび1つのロードバランサに増大した。図7は、結果として得られた論理接続性を示している。

【0063】

制御プレーンが、VSF2という名前で、2つのウェブサーバと1つのロードバランサを必要とする別のVSFを作成する要求を受信するものとする。制御プレーンはCPUEをロードバランサ/ファイアウォールとなるよう割り当て、且つCPUFおよびCPUGを

ウェブサーバとなるよう割り当てる。再び負荷バランシングする2つのコンピューティング要素としてのCPUFおよびCPUGについて知るため、CPU Eを構成する。

【0064】

この構成を実施するため、制御プレーンは、VLAN1にポートv10およびv11が含まれ(つまり、インターネット106に接続)、且つVLAN3にポートv12、v13、v14、v15が含まれるようVLANスイッチ504を構成する。同様に、SANゾーン2にSANポートs6、s7、s9が含まれるようSANスイッチ506を構成する。このSANゾーンは、CPU Eをロードバランサとして、且つCPUFおよびCPUGをSANゾーンのディスクD2に含まれる共有読み取り専用ディスク部分を使用するウェブサーバとして実行させるのに必要なソフトウェアを含む記憶装置を含んでいる。

10

【0065】

図8は、結果として得られる論理接続性のブロック図である。2つのVSF(VSF1、VSF2)が同一の物理VLANスイッチおよびSANスイッチを共有するが、2つのVSFは論理的に分割されている。CPU

B、C、DにアクセスするユーザまたはVSF1を所有または運営する企業は、VSF1のCPUおよびストレージにアクセスできるのみである。このようなユーザはVSF2のCPUまたはストレージにアクセスできない。これは、唯一の共有セグメント(VLAN1)上の別個のVLANおよび2つのファイアウォールの組合せ、および2つのVSFが構成される異なるSANゾーンのために、このようなアクセスができない。

【0066】

さらに、制御プレーンは、VSF1を2つのウェブサーバに戻すことができると判断するものとする。これは、VSF1の負荷の一時的上昇が低下し、あるいはその他の管理行為がとられたためである。それに応じて、制御プレーンは、CPUの電源オフを含む特殊コマンドによってCPUDをシャットダウンする。CPUがシャットダウンすると、制御プレーンはポートv8およびv9をVLAN2から取外し、またはSANゾーン1からSANポートs4と取り外す。ポートs4はアイドルSANゾーンに配置される。アイドルSANゾーンは、例えば、(アイドル用)SANゾーン1またはゾーン0に指定される。

20

【0067】

その後、制御プレーンは別のノードをVSF2に追加することを決定する。これは、VSF2におけるウェブサーバの負荷が一時的に上昇したり、あるいは他の理由によるためである。従って、制御プレーンは、破線経路802で示すように、CPUDをVSF2に配置することを決定する。そのために、VLAN3にポートv8およびv9が含まれ、且つSANゾーン2にSANポートs4が含まれるようVLANスイッチを構成する。CPU Dは、VSF2のサーバに必要なOSおよびウェブサーバソフトウェアの起動用イメージを含むディスク装置2の記憶部分に向けられる。また、CPUDは、VSF2のほかのウェブサーバが共有するファイルシステムのデータへの読み取り専用アクセスが許可される。CPUDは再び電源が投入され、VSF2における負荷バランシングされたウェブサーバとして実行し、SANゾーン1におけるデータまたはVLAN2に取り付けられたCPUへアクセスすることはない。特に、CPUDは、VSF1の一部であった初期の時点でも、VSF1の要素にアクセスすることはできない。

30

40

【0068】

さらに、この構成において、CPU Eによって実行される安全境界は、CPUDを含むまで動的に拡張した。従って、実施例によって、VSFに追加または除去されるコンピューティング要素を適切に保護するように自動的に調整する動的ファイアウォールが提供される。

【0069】

説明のため、実施例はポートに基づくSANゾーニングについて説明した。他の種類のSANゾーニングも用いることができる。例えば、LUNレベルSANゾーニングを使用し、ディスクアレイ内の論理量に基づいてSANゾーンを作成してもよい。LUNレベルSANゾーニングに適した実例製品は、EMC CorporationのVolume

50

Logics Productである。

[0070]

SAN上のディスク装置

起動、あるいは他の共有する必要のあるディスクストレージ、起動プログラムおよびデータをどこで見つけるのかに関する情報を有するディスクストレージへのアクセスという目的で、CPUをSAN上の特定の装置に向ける方法は幾つかある。

[0071]

1つの方法では、コンピューティング要素に取り付けられたSCSI対ファイバーチャネルブリッジング機器およびローカルディスクのSCSIインタフェースを設ける。そのSCSIポートからファイバーチャネルSANの適切な機器への経路を決定することにより、コンピュータは、ローカルに取り付けられたSCSI機器にアクセスするようにファイバーチャネルSAN上の記憶装置にアクセスできる。従って、起動ソフトウェアなどのソフトウェアは、ローカルに取り付けられたSCSI機器をブートオフするように、SAN上のディスク装置を単純にブートオフする。

[0072]

別の方法は、ノードのファイバーチャネルインタフェースおよび関連するデバイスドライバを有し、ファイバーチャネルインタフェースをブート機器として使用可能にするROMおよびOSソフトウェアをブートすることである。

[0073]

他の方法では、SCSIまたはIDE機器コントローラとなるが、SAN上で通信を行なうディスクにアクセスするインタフェースカード（例えばPCIバスまたはSバス）を有する。Solarisなどのオペレーティングシステムは、この方法で使用可能なディスクレスブート機能を完全提供する。

[0074]

通常は、あるノードに関連するSANディスク機器は2種類ある。一方の種類は、他のコンピューティング要素と論理的に共有せず、起動可能OSイメージ、ローカル構成ファイルなどを含む通常はノードごとのルートパーティションであるものを構成する。これは、Unix（登録商標）システム上のルートファイルシステムと同等である。

[0075]

2番目の種類のディスクは、他のノードとの共有ストレージである。共有の種類は、CPU上で実行するOSソフトウェアおよび共有ストレージにアクセスするノードのニーズによって異なる。OSが多数のノード間で共有ディスクパーティションの読み取り/書き込みアクセスを可能にするクラスタファイルシステムを提供する場合、共有ディスクはこのようなクラスタファイルシステムとして実装される。同様に、システムは、共有ディスクへの同時読み取り/書き込みアクセスを行なうために、クラスタ内での多数のノードの実行を可能にするオラクルパラレルサーバなどのデータベースソフトウェアを使用してもよい。このような場合、共有ディスクは、基本OSおよびアプリケーションソフトウェア内にすでに設計されている。

[0076]

このような共有アクセスが不可能であるオペレーティングシステムの場合、OSおよび関連アプリケーションが他のノードと共有するディスク機器を管理できないため、共有ディスクを読み出し専用機器として実装することができる。多数のウェブアプリケーションの場合、ウェブ関連ファイルへ読み出し専用アクセスすればよい。例えば、Unix（登録商標）システムの場合、特定のファイルシステムを読み出し専用として実装してもよい。

[0077]

マルチスイッチコンピューティンググリッド

図5に関連して上記に説明した構成は、複数のVLANスイッチを相互接続して大きな交換VLAN構造を形成することにより、且つ多数のSANスイッチを相互接続して大きな交換SANメッシュを構成することにより、多数のコンピューティングおよび記憶ノードに拡張することができる。この場合、コンピューティンググリッドは、SAN/VLAN

10

20

30

40

50

交換メッシュがCPUおよび記憶装置の非常に多数のポートを含むことを除いて、図5に一般に示すアーキテクチャを有している。制御プレーンを実行する多数のコンピューティング要素は、以下に説明するように、VLAN/SANスイッチの制御ポートに物理的に接続可能である。多数のVLANスイッチを相互接続して複雑な多構内データネットワークを生成することは、この分野において知られている。例えば、G. Haviland による "Designing High-Performance Campus Intrarets with Multilayer Switching (多層切り替えを有する高性能構内イントラネットの設計)" Cisco Systems, Inc., および Brocade から入手可能な情報を参照すること。

【0078】

SANアーキテクチャ

説明では、SANがファイバチャネルスイッチおよびディスク機器、および潜在的にSCSI対ファイバチャネルブリッジなどのファイバチャネルエッジ機器とから構成されることを前提としている。しかし、SANはギガビットイーサネット（登録商標）スイッチなどのほかの技術、または他の物理層プロトコルを使用するスイッチを使用して構成してもよい。特に、IP上でSCSIプロトコルを実行させることにより、IPネットワーク上でSANを構築しようという試みが行なわれている。上述した方法およびアーキテクチャは、これらの他のSAN構築方法に適応できる。VLAN可能層2環境でIP上でSCSIなどのプロトコルを実行させることによってSANを構築する場合、SANゾーンはこれらを異なるVLANにマッピングすることによって生成される。

【0079】

さらに、高速イーサネット（登録商標）またはギガビットイーサネット（登録商標）などのLAN技術上で動作するネットワークアタッチドストレージ（NAS）を使用してもよい。この選択肢により、保全性とコンピューティンググリッドの論理パーティショニングを強化するために、SANゾーンの代わりに異なるVLANを使用する。このようなNAS機器は通常、SunのNSFプロトコルやMicrosoftのSMBなどのネットワークファイルシステムをサポートして、多数のノードが同一のストレージを共有できるようにする。

【0080】

制御プレーンの実施

ここに述べるように、制御プレーンは、SANおよびVLANスイッチの制御およびデータポートに接続される一又は二以上の処理リソースとして実施してもよい。様々な制御プレーンの実施を行なうことができ、且つ本発明は特定の制御プレーンの実施に制限されるものではない。制御プレーン実施の様々な面を、以下の項1) 制御プレーンアーキテクチャ、2) マスターセグメントマネジャー選択、3) 管理機能、4) 方針および保全に関する考察で詳細に説明する。

【0081】

1. 制御プレーンアーキテクチャ

一実施例によれば、制御プレーンは制御プロセス階層として実施される。制御プロセス階層は一般に、一又は二以上のスレーブセグメントマネジャー機構に通信接続されてこれらを制御する一又は二以上のマスターセグメントマネジャー機構を含んでいる。一又は二以上のスレーブセグメントマネジャー機構は、一又は二以上のファームマネジャーを制御する。一又は二以上のファームマネジャーは、一又は二以上のVSFを管理する。マスターおよびスレーブセグメントマネジャー機構は、ハードウェア回路、コンピュータソフトウェア、または何れかの組合せにおいて実施されてもよい。

【0082】

図9は、一実施例による制御プレーン902およびコンピューティンググリッド904との間の論理関係を示すブロック図900である。制御プレーン902は、コンピューティンググリッド904におけるネットワークおよび記憶要素の特殊制御ポートまたはインタフェースを介して、コンピューティンググリッド904に含まれるコンピューティン

10

20

30

40

50

グ、ネットワークングおよび記憶要素を制御および管理する。コンピューティンググリッド904は、上述した実施例により生成された多数のVSF906または論理リソースグループを含む。

#### [0083]

一実施例によると、制御プレーン902はマスターセグメントマネジャー908、一又は二以上のスレーブセグメントマネジャー910、および一又は二以上のファームマネジャー912を含んでいる。マスターセグメントマネジャー908、スレーブセグメントマネジャー910およびファームマネジャー912は、特定のコンピューティングプラットフォーム上の同一位置に配置されたり、あるいは多数のコンピューティングプラットフォーム上で分散されてもよい。便宜上、単一のマスターセグメントマネジャー910のみを図示および説明するが、多数のマスターセグメントマネジャー908を使用してもよい。

10

#### [0084]

マスターセグメントマネジャー908は、スレーブセグメントマネジャー910に通信接続され、これを制御および管理している。各スレーブセグメントマネジャー910は、一又は二以上のファームマネジャー912に通信接続され、これを管理する。一実施例によれば、各ファームマネジャー912は、通信接続された対応するスレーブセグメントマネジャー910として同一のコンピューティングプラットフォーム上の同一位置に配置される。ファームマネジャー912は、コンピューティンググリッド904上でVSF906を確立、構成および維持する。一実施例によれば、各ファームマネジャー912は管理する単一のVSF906が割り当てられるが、ファームマネジャー912も多数のVSF906が割り当てられる。ファームマネジャー912はそれぞれ直接ではなく、各スレーブセグメントマネジャー910を介してのみ通信を行う。スレーブセグメントマネジャー910は、その割り当てられたファームマネジャー912の状態を監視する。スレーブセグメントマネジャー910は、機能停止や異常終了したそれぞれ割り当てられたファームマネジャー912を再開させる。

20

#### [0085]

マスターセグメントマネジャー908はVSF906のローディングを監視して、各VSF906に割り当てるリソースの量を決定する。マスターセグメントマネジャー908は、必要に応じてファームマネジャー912を介してVSFのリソースを割り当ておよび割り当て解除するようにスレーブセグメントマネジャー910に指示する。特定のアプリケーションの必要条件に応じて様々な負荷バランシングアルゴリズムを実施してもよく、且つ本発明は特定の負荷バランシング方法に限定されるものではない。

30

#### [0086]

マスターセグメントマネジャー908は、スレーブセグメントマネジャー910およびファームマネジャー912が実行されているコンピューティングプラットフォームのローディング情報を監視して、コンピューティンググリッド904は適切にサービスされているかを判断する。マスターセグメントマネジャー908はスレーブセグメントマネジャー910の割り当ておよび割り当て解除を行い、必要に応じてコンピューティンググリッド904を適切に管理するためにファームマネジャー912の割り当ておよび割り当て解除を行うようスレーブセグメントマネジャー910を指示する。一実施例によれば、マスターセグメントマネジャー908も、必要に応じてファームマネジャー912およびスレーブセグメントマネジャー910の間で負荷をバランスさせるために、ファームマネジャー912へのVSFの割り当て、およびスレーブセグメントマネジャー910へのファームマネジャー912の割り当てを管理する。一実施例によれば、スレーブセグメントマネジャー910はマスターセグメントマネジャー908と活発に通信し、コンピューティンググリッド904への変更要求、および別スレーブセグメントマネジャー910および/またはファームマネジャー912の要求を行う。一又は二以上のスレーブセグメントマネジャー910および一又は二以上のファームマネジャー912を実行している処理プラットフォームが機能しなくなった場合、マスターセグメントマネジャー908は、停止したコンピューティングプラットフォームのファームマネジャー912から他のファームマ

40

50

ネジャー 912へVSF906を再割り当てする。この場合、マスターセグメントマネジャー 908も、VSF906の再割り当てを行うために別のファームマネジャー 912を開始するようにスレーブセグメントマネジャー 910に指示することができる。VSF906に割り当てられた多数のコンピューティングリソース、多数のアクティブなファームマネジャー 912、およびスレーブセグメントマネジャー 910をアクティブに管理することにより、全体的な電力消費量を制御できる。例えば、電力を節約するために、マスターセグメントマネジャー 908は、アクティブなスレーブセグメントマネジャー 910またはファームマネジャー 912を有していないコンピューティングプラットフォームをシャットダウンしてもよい。節電は、大きなコンピューティンググリッド904および制御プレーン902で重要となる。

10

#### 【0087】

一実施例によれば、マスターセグメントマネジャー 908は、レジストリを使用することでスレーブセグメントマネジャー 910を管理する。レジストリは、その状態、割り当てられたファームマネジャー 912、および割り当てられたVSF906などの現在のスレーブセグメントマネジャー 910についての情報を含んでいる。スレーブセグメントマネジャー 910が割り当ておよび割り当て解除されると、レジストリは更新されて、スレーブセグメントマネジャー 910の変更が反映される。例えば、新しいスレーブセグメントマネジャー 910がマスターセグメントマネジャー 908および割り当てられた一又は二以上のVSF906によって例示化されると、レジストリが更新されて、新しいスレーブセグメントマネジャー 910およびその割り当てられたファームマネジャー 912とVSF906の生成が反映される。次に、マスターセグメントマネジャー 908はレジストリを定期的に調べて、スレーブセグメントマネジャー 910へどのようにVSF906を割り当てるのがよいのかを判断することができる。

20

#### 【0088】

一実施例によれば、レジストリは、マスターセグメントマネジャー 910がアクセスできるマスターセグメントマネジャー 908についての情報を含んでいる。例えば、レジストリは一又は二以上のアクティブなマスターセグメントマネジャー 908を識別するデータを含んでいてもよい。新しいスレーブセグメントマネジャー 910が生成されると、新しいスレーブセグメントマネジャー 910はレジストリをチェックして、一又は二以上のマスターセグメントマネジャー 908の識別について確認することができる。

30

#### 【0089】

レジストリは様々な形で実施されてもよく、且つ本発明は特定の実施方法に限定されない。例えば、レジストリは制御プレーン902内のデータベース914に保存されるデータファイルであってもよい。レジストリは、制御プレーン902の外に保存されなくてもよい。例えば、レジストリはコンピューティンググリッド904の記憶装置に保存されてもよい。この例では、記憶装置は制御プレーン902専用となり、VSF906に割り当てられない。

#### 【0090】

##### 2. マスターセグメントマネジャー選出

一般に、マスターセグメントマネジャーは、制御プレーンが確立されたとき、あるいは既存のマスターセグメントマネジャーが故障した後に、選出される。一般に特定の制御プレーンに対して単一のマスターセグメントマネジャーが存在するが、2つ以上のマスターセグメントマネジャーを選出して、制御プレーンのスレーブセグメントマネジャーを同時管理するほうが有利な場合もある。

40

#### 【0091】

一実施例によれば、制御プレーンにおけるスレーブセグメントマネジャーは、その制御プレーンのマスターセグメントマネジャーを選出する。マスターセグメントマネジャーがなく、単一のスレーブセグメントマネジャーのみが存在するという単純なケースでは、スレーブセグメントマネジャーがマスターセグメントマネジャーとなり、必要に応じて別のスレーブセグメントマネジャーを割り当てる。2つ以上のスレーブセグメントマネジャーが

50

存在する場合、2つ以上のスレーブプロセスが例えば定足数などの採決によって新しいマスターセグメントマネジャーを選出する。

【0092】

制御ブレーンのスレーブセグメントマネジャーは必ずしも永続的ではないので、特定のスレーブセグメントマネジャーを選択して、採決に参加させてもよい。例えば、一実施例によれば、レジスタは、各スレーブセグメントマネジャーによって周期的に更新される各スレーブセグメントマネジャーのタイムスタンプを含んでいる。指定された選択基準に従って決定された、最も最近に更新されたタイムスタンプを有するスレーブセグメントマネジャーはいまだに実行されていると考えられ、新しいマスターセグメントマネジャーを選出するために選択される。例えば、指定数の最も新しいスレーブセグメントマネジャーを採決に選択してもよい。

10

【0093】

一実施例によれば、選出シーケンス番号を全てのアクティブなスレーブセグメントマネジャーに割り当て、アクティブなスレーブセグメントマネジャーの選出シーケンス番号に基づいて新しいマスターセグメントマネジャーを決定する。例えば、最も低いあるいは最も高い選出シーケンス番号を使用して、特定のスレーブセグメントマネジャーを次の（または最初の）マスターセグメントマネジャーに選択してもよい。

【0094】

マスターセグメントマネジャーが確立されると、マスターセグメントマネジャーとしての同一制御ブレーンのスレーブセグメントマネジャーは、現在のマスターセグメントマネジャーにコンタクト（ピング）することによりマスターセグメントマネジャーの検診を周期的に行って、マスターセグメントマネジャーがまだアクティブであるか否かを判断する。現在のマスターセグメントマネジャーがアクティブでないと判断した場合、新しいマスターセグメントマネジャーを選出する。

20

【0095】

図10は、実施例によるマスターセグメントマネジャー選出の状態図1000を示している。スレーブセグメントマネジャーのメインループである状態1002において、スレーブセグメントマネジャーは、ピングタイマーの終了を待つ。ピングタイマーが終了すると、状態1004となる。状態1004において、スレーブセグメントマネジャーは、マスターセグメントマネジャーをピングする。さらに、状態1004において、スレーブセグメントマネジャーのタイムスタンプ（TS）が更新される。マスターセグメントマネジャーがピングに応答した場合、マスターセグメントマネジャーはまだアクティブであり、状態1002に戻る。特定時間後もマスターセグメントマネジャーから応答がなければ、状態1006になる。

30

【0096】

状態1006において、アクティブなスレーブセグメントマネジャーのリストを得て、状態1008になる。状態1008において、他のスレーブセグメントマネジャーもマスターセグメントマネジャーからの応答を受信していないか確認する。この確認を行うためにスレーブセグメントマネジャーへメッセージを送る代わりに、この情報をデータベースから得る。マスターセグメントマネジャーがアクティブでないことにスレーブセグメントマネジャーが同意しない、すなわち又は二以上のスレーブセグメントマネジャーがマスターセグメントマネジャーから適時の応答を受信した場合、現在のマスターセグメントマネジャーがまだアクティブであると推定され、状態1002に戻る。特定の数のスレーブセグメントマネジャーが現在のマスターセグメントマネジャーから適時の応答を受信しなかった場合、現在のマスターセグメントマネジャーが「死んでいる」、すなわちアクティブでないと推定され、状態1010に進む。

40

【0097】

状態1010において、プロセスを開始したスレーブセグメントマネジャーは選出テーブルから現在の選出番号、且つデータベースから次の選出番号を検索する。次に、スレーブセグメントマネジャーは選出テーブルを更新して、次の選出番号と一義的なアドレスを指

50

定するエントリをマスター選出テーブルに書き込む。次に、スレーブセグメントマネジャーが現在の選出番号の最も低いシーケンス番号を読み出す状態1012に進む。状態1014において、特定のスレーブセグメントマネジャーが最も低いシーケンス番号を有しているか否かを確認する。有していない場合、状態1002に戻る。有している場合、特定のスレーブセグメントマネジャーがマスターセグメントマネジャーになる状態1016に進む。次に、状態1018に進み、選出番号をインクリメントする。

【0098】

上述したように、スレーブセグメントマネジャーは一般に、その割り当てられたVSFのサービスと、マスターセグメントマネジャーからの命令に応じて新たなVSFの割り当てを行う。スレーブセグメントマネジャーはまたマスターセグメントマネジャーのチェックと、必要に応じて新たなマスターセグメントマネジャーの選出も行う。

10

【0099】

図11は、実施例によるスレーブセグメントマネジャーの様々な状態を示す状態図1100である。処理は、スレーブセグメントマネジャー開始状態1102において始まる。状態1102から、現在のマスターセグメントマネジャーの状態を確認する要求に応じて、状態1104に進む。状態1104では、スレーブセグメントマネジャーは現在のマスターセグメントマネジャーにピングを送って、現在のマスターセグメントマネジャーがまだアクティブであるか否かを判断する。適時の応答が現在のマスターセグメントマネジャーからあれば、状態1106に進む。状態1106では、他のスレーブセグメントマネジャーにメッセージが同報通信され、マスターセグメントマネジャーがピングに回答したことを知らせる。状態1106から、開始状態1102に戻る。

20

【0100】

状態1104で、適時のマスター応答がなければ、状態1108に進む。状態1108では、他のスレーブセグメントマネジャーにメッセージが同報通信され、マスターセグメントマネジャーがピングに回答しなかったことを知らせる。次に、開始状態1102に戻る。ちなみに、十分な数のスレーブセグメントマネジャーが現在のマスターセグメントマネジャーから応答を受信しなかった場合、新しいマスターセグメントマネジャーを上記のように選出する。

【0101】

状態1102から、マスターセグメントマネジャーからVSFを再開する要求を受信したら、状態1110に進む。状態1110では、VSFが再開されて、開始状態1102に戻る。

30

【0102】

上述したように、マスターセグメントマネジャーは一般に、マスターセグメントマネジャーが制御するコンピューティンググリッドのVSFが一又は二以上のスレーブセグメントマネジャーによって適切にサービスされるようにする。このために、マスターセグメントマネジャーは、マスターセグメントマネジャーとしての同一制御プレーンの全てのスレーブセグメントマネジャーの定期的検診を行う。一実施例によれば、マスターセグメントマネジャー908は、スレーブセグメントマネジャー910から状態情報を周期的に要求する。情報は例えば、どのVSF906がスレーブセグメントマネジャー910によってサービスされているかを含んでいる。特定のスレーブセグメントマネジャー910が特定時間内に応答しなければ、マスターセグメントマネジャー908は特定のスレーブセグメントマネジャー910の再開を試みる。特定のスレーブセグメントマネジャー910によって再開できない場合、マスターセグメントマネジャー908は、異常のあるスレーブセグメントマネジャー910から別のスレーブセグメントマネジャー910にファームマネジャー912を再割り当てする。次に、マスターセグメントマネジャー908は一又は二以上の別のスレーブセグメントマネジャー910を例示化して、プロセスローディングの再バランシングを行うことができる。一実施例によれば、マスターセグメントマネジャー908は、スレーブセグメントマネジャー910を実行しているコンピューティングプラットフォームの状態を監視する。コンピューティングプラットフォームに異常があれば、マスター

40

50

セグメントマネジャー 908 は、異常のあるコンピューティングプラットフォーム上のプラットフォームマネジャー 912 に割り当てられた VSF を、別のコンピューティングプラットフォームに割り当てる。

【0103】

図 12 は、マスターセグメントマネジャーの状態図 1200 である。処理は、マスターセグメントマネジャー開始状態 1202 において開始する。状態 1202 から、マスターセグメントマネジャー 908 が制御面 902 のスレーブセグメントマネジャー 910 の周期的検査を行うかあるいはこれを要求したときに、状態 1204 に進む。状態 1204 から、全てのスレーブセグメントマネジャー 910 が予測したように応答した場合、状態 1202 に戻る。これは、全てのスレーブセグメントマネジャー 910 が、全てのスレーブセグメントマネジャー 910 が普通に動作していることを示す特定の情報をマスターセグメントマネジャー 908 に提供した場合に、生じる。一又は二以上のスレーブセグメントマネジャー 910 が応答しない、あるいは一又は二以上のスレーブセグメントマネジャー 910 に異常があったことを示す応答をした場合、状態 1206 に進む。

【0104】

状態 1206 において、マスターセグメントマネジャー 908 は異常のあったスレーブセグメントマネジャー 910 の再開を試みる。これはいくつかの方法で行なうことができる。例えば、マスターセグメントマネジャー 908 は、応答のないあるいは異常のあったスレーブセグメントマネジャー 910 に再開メッセージを送ることができる。状態 1206 から、全てのスレーブセグメントマネジャー 910 が予想したように応答、すなわち問題なく再開された場合、状態 1202 戻る。例えば、異常のあったスレーブセグメントマネジャー 910 が問題なく再開すると、スレーブセグメントマネジャー 910 はマスターセグメントマネジャー 908 に再開確認メッセージを送る。状態 1206 から、一又は二以上のスレーブセグメントマネジャーが再開できなかった場合、状態 1208 に進む。これは、マスターセグメントマネジャー 908 が特定のスレーブセグメントマネジャー 910 から再開確認メッセージを受信しない場合に生じる。

【0105】

状態 1208 において、マスターセグメントマネジャー 908 は、スレーブセグメントマネジャー 910 を実行するマシンの現在のローディングを決定する。スレーブセグメントマネジャー 908 のローディング情報を得るために、マスターセグメントマネジャー 908 は、スレーブセグメントマネジャー 910 を直接ローディングするか、あるいは例えばデータベース 914 など別の場所からローディング情報を得る。本発明は、マスターセグメントマネジャー 908 がスレーブセグメントマネジャー 910 のローディング情報を得るための特定の方法に限定されない。

【0106】

次に状態 1210 に進み、異常のあったスレーブセグメントマネジャー 910 に割り当てられた VSF 906 を他のスレーブセグメントマネジャー 910 に再割り当てする。VSF 906 が割り当てられているスレーブセグメントマネジャー 910 は、いつ再割り当てが完了したのかをマスターセグメントマネジャー 908 に知らせる。例えば、スレーブセグメントマネジャー 910 はマスターセグメントマネジャー 908 に再割り当て確認メッセージを送って、VSF 906 の再割り当てが問題なく終了したことを知らせることができる。異常のあったスレーブセグメントマネジャー 910 に関連する全ての VSF 906 の再割り当てが確認されるまで、状態 1210 に留まる。確認されれば、状態 1202 に戻る。

【0107】

異常のあったスレーブセグメントマネジャー 910 に関連する VSF 906 を他のアクティブスレーブセグメントマネジャー 910 へ再割り当てする代わりに、マスターセグメントマネジャー 908 は別のスレーブセグメントマネジャー 910 を割り当て、新しいスレーブセグメントマネジャー 910 にこれらの VSF 906 を割り当ててもよい。既存のスレーブセグメントマネジャー 910 または新しいスレーブセグメントマネジャー 910 へ

10

20

30

40

50

V S F 9 0 6を再割り当てするかどうかの選択は、少なくとも部分的に、新しいスレーブセグメントマネジャー910の割り当てに関連する待ち時間、および既存のスレーブセグメントマネジャー910へのV S F 9 0 6の再割り当てに関連する待ち時間に依る。何れの方法も特定のアプリケーションの必要条件に応じて使用することができ、且つ本発明は何れの方法にも限定されることはない。

[0108]

### 3. 管理機能

一実施例によれば、制御ブレン902は、グローバルグリッドマネジャーに通信接続されている。制御ブレン902は、グローバルグリッドマネジャーに、課金、障害、容量、ローディング、および他のコンピューティンググリッド情報を提供する。図13は、実施例によるグローバルグリッドマネジャーの使用を説明するブロック図である。

[0109]

図13において、コンピューティンググリッド1300は、グリッドセグメント1302と呼ばれる論理部分にパーティションされる。各グリッドセグメント1302は、データブレン904を制御および管理する制御ブレン902を含んでいる。この例において、各データブレン904は図9のコンピューティンググリッド904と同一であるが、多数の制御ブレン902およびデータブレン904、すなわちグリッドセグメント1302を管理するグローバルグリッドマネジャーの使用を説明するため、「データブレン」と呼ばれる。

[0110]

各グリッドセグメントは、グローバルグリッドマネジャー1304に通信接続される。グローバルグリッドマネジャー1304、制御ブレン902、およびコンピューティンググリッド904は、単一のコンピューティングプラットフォームに同時配置されたり、あるいは多数のコンピューティングプラットフォーム上で分散させてもよく、本発明は特定の実施方法に限定されることはない。

[0111]

グローバルグリッドマネジャー1304は、複数のグリッドセグメント1302の集中管理およびサービスを行う。グローバルグリッドマネジャー1304は、様々な管理タスクで使用される制御ブレン902からの課金、ローディング、および他の情報を集めることができる。例えば、課金情報を使用して、コンピューティンググリッド904が提供するサービスの課金を行う。

[0112]

### 4. 方針および保全についての考察

上述したように、制御ブレンにおけるスレーブセグメントマネジャーは、コンピューティンググリッドにおける関連するV S Fと通信可能でなければならない。同様に、コンピューティンググリッドにおけるV S Fは、その関連するスレーブセグメントマネジャーと通信可能でなければならない。更に、コンピューティンググリッドにおけるV S Fは、あるV S Fが何らかの方法で他のV S Fの構造を変えてしまうのを防ぐために、互いに通信可能であってはならない。これらの方針を実施する様々な方法について説明する。

[0113]

図14は、実施例によるコンピューティンググリッドへ制御ブレンを接続するアーキテクチャのブロック図1400である。参照番号1402でまとめて識別されるV L A N スイッチ (V L A N SW1~V L A N SWn) および参照番号1404でまとめて識別されるS A N スイッチ (S A N SW1~S A N SWn) の制御 (「C T L」) ポートは、イーサネット (登録商標) サブネット1406に接続される。イーサネット (登録商標) サブネット1406は、参照番号1408でまとめて識別される複数のコンピューティング要素 (C P U 1, C P U 2~C P U n) に接続される。従って、制御ブレン1408のコンピューティング要素のみが、V L A N スイッチ1402およびS A N スイッチ1404の制御ポート (C T L) に通信接続される。この構造は、V S F (図示せず) におけるコンピューティング要素が、それ自身または他のV S Fに関連するV L A Nおよび

SANゾーンのメンバーシップを変更してしまうを防ぐ。この方法も、制御ポートがシリアルまたはパラレルポートである場合に適用可能である。この場合、ポートは制御ブレン1408のコンピューティング要素に接続される。

【0114】

図15は、実施例による制御ブレンコンピューティング要素(CP CPU1、CP CPU2～CP CPU<sub>n</sub>)1502をデータポートに接続する構造を示すブロック図1500である。この構成において、制御ブレンコンピューティング要素502は、制御ブレンコンピューティング要素1502のために動作する制御ブレンエージェント1504に周期的にパケットを送る。制御ブレンエージェント1504は、リアルタイムデータのためにコンピューティング要素502を周期的にポーリングして、データを制御ブレンコンピューティング要素1502に送る。制御ブレン1502における各セグメントマネージャは、制御ブレン(CP)LAN1506に通信接続されている。CP LAN1506は、CPファイアウォール1508を介して、VLANスイッチ504の特殊ポートV17に通信接続されている。この構造により、制御ブレンコンピューティング要素1502に拡張可能な確実な手段が与えられ、コンピューティング要素502からリアルタイム情報が集められる。

【0115】

図16は、実施例によるコンピューティンググリッドへ制御ブレンを接続するアーキテクチャのブロック図1600である。制御ブレン1602は、制御ブレンコンピューティング要素CP CPU1、CP CPU2～CP CPU<sub>n</sub>を含んでいる。制御ブレン1602における各制御ブレンコンピューティング要素CP CPU1、CP CPU2～CP CPU<sub>n</sub>は、全体でSANメッシュ1604を形成する複数のSANスイッチのポートS1、S2～S<sub>n</sub>に通信接続される。

【0116】

SANメッシュ1604は、制御ブレン1602に対してプライベートであるデータを含む記憶装置1606に通信接続されるSANポートS<sub>o</sub>、S<sub>p</sub>を含んでいる。記憶装置1606は、便宜上、ディスクとして図16に示されている。記憶装置1606は、いずれのタイプの記憶媒体で実施されてもよく、本発明は記憶装置1606の特定の種類の記憶媒体に限定されることはない。記憶装置1606は、制御ブレンプライベート記憶ゾーン1608に論理的に配置される。制御ブレンプライベート記憶ゾーン1608は、制御ブレン1602を実施するログファイル、統計データ、現在の制御ブレン構成情報を維持する。SANポートS<sub>o</sub>、S<sub>p</sub>は制御ブレンプライベート記憶ゾーンの唯一の部分であり、他のSANゾーンには配置されることはないため、制御ブレン1602におけるコンピューティング要素のみが記憶装置1606にアクセスできる。また、S1、S2～S<sub>n</sub>、S<sub>o</sub>およびS<sub>p</sub>は、制御ブレン1602におけるコンピューティング要素に通信接続されるのみ制御ブレンSANゾーンに存在する。これらのポートは、VSFにおけるコンピューティング要素(図示せず)がアクセスすることはできない。

【0117】

一実施例によれば、特定のコンピューティング要素CP CPU1、CP CPU2～CP CPU<sub>n</sub>が記憶装置またはその一部にアクセスする必要がある場合、それは特定のVSFの一部であり、特定のコンピューティング要素は特定のVSFのSANゾーンに置かれる。例えば、コンピューティング要素CP CPU2がVSF i ディスク1610にアクセスする必要があるものとする。この場合、制御ブレンCP CPU2に関連するポートs2は、ポートSiを含むVSF iのSANゾーンに配置される。一度コンピューティング要素CP CPU2がポートSiのVSF i ディスク1610へアクセスすると、コンピューティング要素CP CPU2はVSF iのSANゾーンから取り除かれる。

【0118】

同様に、コンピューティング要素CP CPU1がVSF j ディスク1612にアクセスする必要があるものとする。この場合、コンピューティング要素CP CPU1はVSF jに関連するSANゾーン内に配置される。その結果、ポートS1は、ポートSjを含む

10

20

30

40

50

ゾーンを有するVSSFjに関連するSANゾーン内に配置される。一度コンピューティング要素CP CPU1がポートSjに接続されたVSSFjディスク1612へアクセスすると、コンピューティング要素CP CPU1はVSSFjに関連するSANゾーンから除去される。この方法により、正確なSANゾーン制御を使用してリソースへのアクセスを正確に制御することによる、制御ブレンコンピューティング要素および制御ブレン記憶ゾーン1608の完全性が得られる。

#### [0119]

上述したように、単一の制御ブレンコンピューティング要素は複数のVSSFの管理を行うことができる。従って、単一の制御ブレンコンピューティング要素は、各制御ブレンに大して確立された方針規則に従ってVSSF間のファイアウォールを実行しながら、多数のVSSFにおける自身を同時に明確にできなければならない。方針規則は、各制御ブレンのデータベース914 (図9) に保存、あるいは中央セグメントマネジャー1302 (図13) によって実施してもよい。

#### [0120]

一実施例によれば、(物理的スイッチ) ポートに基づくVLANタグはスプーフできないため、VLANタグとIPアドレスとの間を強固に結合させて、VSSFによるスプーフ攻撃を防いでいる。あるVLANインタフェースで送られてくるIPパケットは、パケットが到着する論理インタフェースと同じVLANタグおよびIPアドレスを有していなければならない。これにより、VSSFにおける不正サーバが別のVSSFにおけるソースIPアドレスをスプーフし、別のVSSFの論理構造を潜在的に変更し、あるいはコンピューティンググリッド機能の保全を破壊するIPスプーフ攻撃を防止する。このVLANタグを防止するほう方法では、高安全(クラスA) データセンターを使用して防止できるコンピューティンググリッドへの物理的アクセスが必要である。

#### [0121]

様々なネットワークフレームタグ形式を使用してデータパケットのタグを行ってもよく、且つ本発明は特定のタグ形式に限定されることはない。一実施例によれば、他の形式も適切であるが、IEEE802.1qのVLANタグを使用している。この例では、VLAN/IPアドレス一貫性チェックを、アクセスを制御するために802.1qタグ情報が存在するIPスタックのサブシステムで実行する。この例において、コンピューティング要素は、コンピューティング要素が多数のVLANに同時に通信接続されるよう、VLAN可能ネットワークインタフェースカード(NIC) で構成されている。

#### [0122]

図17は、実施例によるVLANタグとIPアドレスとの間を強固に結合する構成のブロック図1700である。コンピューティング要素1702および1704は、NIC1708および1710を介して、VLANスイッチ1706のポートv1およびv2にそれぞれ通信接続される。VLANスイッチ1706も、アクセススイッチ1712および1714に通信接続される。ポートv1およびv2は、タグ形式で構成される。一実施例によれば、IEEE802.1qのVLANタグ情報は、VLANスイッチ1706によって提供される。

#### [0123]

広域コンピューティンググリッド

上述したVSSFは、様々な方法でWAN上に分散される。

#### [0124]

一つの方法では、広域バックボーンは、非同期転送モード(ATM) 切替に基づいていてもよい。この場合、各ローカルエリアVLANは、ATM LANエミュレーション(LANE) 標準の一部であるエミュレートッドLAN (ELAN) を使用して広域に拡張される。このように、単一のVSSFは、ATM/SONET/OC-12リンクなどの幾つかの広域リンク全体に広がる。ELANは、ATM WAN全体に拡張するVLANの一部となる。

#### [0125]

10

20

30

40

50

他の方法では、VSFをVPNシステムを使用してWAN全体に拡張する。本実施例において、ネットワークの根本的特徴は不適切になり、VPNを使用して2つ以上のVSFをWAN全体にわたって相互接続し、単一の分散VSFを生成する。

【0126】

分散VSFにおいてデータを論理コピーするために、データミラーリング技術を使用することができる。あるいは、SAN対ATMブリッジングまたはSAN対ギガビットイーサネット（登録商標）ブリッジングなどの幾つかのSAN対WANブリッジング技術のうちの1つを使用して、WAN上にSANをブリッジさせる。IPはこのようなネットワーク上で問題なく動作するので、IPネットワーク上に構成されたSANはWAN上で自然に拡張する。

【0127】

図18は、WAN接続上で拡張した複数のVSFのブロック図である。サンノゼセンター、ニューヨークセンター、およびロンドンセンターは、WAN接続によって接続されている。各WAN接続は、上述したようにATM、ELANまたはVPN接続から構成される。各センターは、少なくとも1つのVSFおよび少なくとも1つのアイドルプールから構成される。例えば、サンノゼセンターはVSF1AおよびアイドルプールAを有している。この構成において、センターの各アイドルプールのコンピューティングリソースは、他のセンターにあるVSFへの割り当てまたは指定に対して利用できる。このような割り当てまたは指定が行われると、VSFはWAN上で拡張する。

【0128】

VSFの使用例

上記例で説明したVSFアーキテクチャは、ウェブサーバシステムのからみで使用してもよい。従って、上記例は、特定のVSFにおけるCPUから構成したウェブサーバ、アプリケーションサーバおよびデータベースサーバに関して説明した。しかし、VSFアーキテクチャを他の多くのコンピューティング状況で使用し、他の種類のサービスを提供してもよく、且つ本発明はウェブサーバシステムに限定されるものではない。

【0129】

一内容分散ネットワークの一部としての分散VSF

一実施例において、VSFは、広域VSFを使用して内容分散ネットワーク（CDN）を提供する。CDNは、データの分散キャッシングを行うキャッシングサーバのネットワークである。キャッシングサーバのネットワークは、例えば、Inktomi Corporation、San Mateo, Californiaから販売されているTraffic Server（TS）ソフトウェアを使用して実施できる。TSはクラスウェアシステムであり、システムは、更に多くのCPUがキャッシングトラフィックサーバコンピューティング要素の集合に追加されると、拡張する。従って、CPUの追加が拡張の機構であるシステムに非常に適している。

【0130】

この構成において、システムは、TSなどのキャッシングソフトウェアを実行するVSFの部分に更に多くのCPUを動的に追加できるので、バースト状のウェブトラフィックが生じるのに近い地点でキャッシュ容量を増大させることが可能である。その結果、CDNは、適法的な方法でCPUおよびI/O帯域幅において動的に拡張するように構成される。

【0131】

一ホステッドイントラネットアプリケーションのVSF

ホストおよび管理されたサービスとして、企業リソースプランニング（ERP）、ORMおよびCRMソフトウェアなどのイントラネットアプリケーションの提供への興味が増大している。Citrix WinFrameおよびCitrix MetaFrameなどの技術により、企業は、Windows（登録商標）CE機器またはウェブブラウザなどの小型軽量クライアント上のサービスとしてMicrosoft Windows（登録商標）アプリケーションを提供することができる。VSFは拡張可能にこのようなアプリケーションをホストすることが可能である。

10

20

30

40

50

## 【0132】

例えば、ドイツのSAP Aktiengesellschaftより販売されているSAP R/3 ERPソフトウェアにより、企業は多数のアプリケーションおよびデータサーバを使用してバランスをロードさせることができる。VSFの場合、リアルタイムの要求または他の要因に基づいてVSFを拡張するために、企業は更に多くのアプリケーションサーバ（例えば、SAPダイアログサーバ）をVSFに動的に追加する。

## 【0133】

同様に、Citrix Metaframeにより、更に多くのCitrixサーバを追加することにより、ホステッドWindows（登録商標）アプリケーションを実行するサーバファーム上でWindows（登録商標）アプリケーションユーザを拡張することができる。この場合、VSFに対し、Citrix MetaFrame VSFは、更に多くのMetaframeがホストするWindows（登録商標）アプリケーションのユーザを収容するために更に多くのCitrixサーバを動的に追加する。多くのほかのアプリケーションが上述した例と同様にホストされることが明らかとなる。

## 【0134】

—VSFとの顧客相互作用

VSFは求めに応じて生成されるため、VSFを「所有する」VSF顧客または組織は、VSFをカスタマイズするために様々な方法でシステムと互いに影響し合うことができる。例えば、VSFは制御プレーンを介して即座に生成および変更されるので、VSF顧客は特権アクセスが許されて、そのVSF自身を生成および変更してもよい。特権アクセスは、ウェブページおよび保安アプリケーション、トークンカード認証、ケルベロス交換、または他の適切な保安要素によって与えられたパスワード認証を使用して与えられる。

## 【0135】

一実施例において、一式のウェブページは、コンピューティング要素または別個のサーバによって供給される。ウェブページにより、顧客は、層の数、特定の層におけるコンピューティング要素の数、各要素に対して使用されるハードウェアおよびソフトウェアプラットフォーム、どの種類のウェブサーバ、アプリケーションサーバ、またはデータベースサーバソフトウェアこれらのコンピューティング要素上で事前に構成するかなどを指定することによって、カスタムVSFを生成することができる。従って、顧客は仮想供給コンソールを備えている。

## 【0136】

顧客またはユーザがこのような供給情報を入力した後、制御プレーンはオーダーを解析および評価し、それを実行するために待ち行列に入れる。オーダーは人間の管理者が再検討して、適切であることを確認することができる。企業のクレジット確認を実行させて、要求されたサービスに対して支払いを行う適切なクレジットを有していることを確認できる。供給オーダーが承認されると、制御プレーンは順序に適合するVSFを構成し、VSFにおける一又は二以上のコンピューティング要素へのルートアクセスを与えるパスワードを顧客に返す。次に、顧客はアプリケーションのマスターコピーをアップロードして、VSFで実行することができる。

## 【0137】

コンピューティンググリッドを採用する企業が営利目的の企業である場合、ウェブページから、クレジットカード、P.O.番号、電子小切手、または他の支払方法などの支払いに関する情報も受信することができる。

## 【0138】

別の実施例において、ウェブページにより、顧客は、リアルタイムロードに基づいて、要素の最小数と最大数との間のVSFの自動拡大縮小など、幾つかのVSFサービスプランのうちの1つを選択することができる。顧客は、ウェブサーバなどの特定の層におけるコンピューティング要素の最小数、またはVSF最小サーバ容量を有していなければならない期間などのパラメータの変更を可能にする制御値を有することができる。パラメータは、顧客の為替手形割引率を自動的に調整し、且つ課金ログファイル項目を生成する課金ソ

10

20

30

40

50

フトウェアにリンクしていてもよい。

#### 【0139】

特権アクセス機構により、顧客は報告書を得て、使用、ロード、毎秒のヒット数またはトランザクション数に関するリアルタイム情報を監視し、リアルタイム情報に基づくV S Fの特徴を調整することができる。上記特色により、サーバファームの構築に対する従来の手動による方法よりも優れた利点が得られる。従来の方法では、ユーザは、様々な方法でサーバを追加し、サーバファームを構成する面倒な手順を介さずに、サーバファームの特性を自動的に変更することはできない。

#### 【0140】

##### ーV S Fに対する課金モデル

V S Fの動的性質を考えると、コンピューティンググリッドおよびV S Fを採用する企業は、V S Fのコンピューティング要素および記憶要素の実際の使用に基づくV S Fの課金モデルを使用して、V S Fを所有する顧客に対してサービス料金を請求することができる。ここに開示するV S Fアーキテクチャおよび方法は、あるV S Fのリソースは静的に指定されないで、「即金払い」課金モデルを可能にする。従って、そのサーバファームの使用負荷が極めて変わりやすい特定の顧客は、一定のピークサーバ容量に関連する料金は課金されず、使用、瞬間使用などの実行平均を反映する料金が課金されるので、料金を節約することができる。

#### 【0141】

例えば、企業は、10台のサーバなどのコンピューティング要素の最小数に対する均一料金を規定し、且つリアルタイムの負荷が10以上の要素を必要としたときを規定する課金モデルを使用して運営するので、ユーザは、何台の追加サーバが必要であり、且つそれらが必要であった時間に基づいて、追加サーバの追加料金が課金される。このような課金の単位は、請求されるリソースを反映してもよい。例えば、課金は、M I P S時間、C P U時間、C P U千秒などの単位で表してもよい。

#### 【0142】

##### ー顧客可視制御プレーンA P I

他の方法では、V S Fの容量は、リソース変更のための制御プレーンの呼び出しを規定するアプリケーションプログラミングインタフェース (A P I) を顧客に与えることで、制御されてもよい。従って、顧客が用意したアプリケーションプログラムは、A P Iを使用して呼び出しまたは要求を発生し、更に多くのサーバ、更に多くのストレージ、更に高い処理能力などを要求することができる。この方法は、顧客がコンピューティンググリッド環境について知り、制御プレーンが与える能力を利用するためにアプリケーションプログラムを必要とするときに使用してもよい。

#### 【0143】

上記アーキテクチャにおいて、何れの部分も、顧客がコンピューティンググリッドとの使用でそのアプリケーションを変更する必要はない。既存のアプリケーションは、手動構成したサーバファームで動作すると同様に動作する。しかしながら、制御プレーンによって与えられるリアルタイム負荷監視機能に基づいて必要とするコンピューティングリソースをよりよく理解するのであれば、アプリケーションはコンピューティンググリッドで可能なダイナミズムを利用することができる。アプリケーションプログラムによるサーバファームのコンピューティング容量の変更を可能にする上記性質のA P Iは、サーバファームの構築に対する既存の手動方法を用いては可能ではない。

#### 【0144】

##### ー自動更新およびバージョンing

ここに開示する方法および機構を使用し、制御プレーンは、V S Fのコンピューティング要素で実行されるオペレーティングシステムソフトウェアの自動更新およびバージョンingを行うことができる。従って、エンドユーザまたは顧客は、新たなパッチ、バグフィックスなどでオペレーティングシステムを更新することについて心配する必要はない。制御プレーンは、このようなソフトウェア要素が受信されるとそのライブラリを維持し、影響

10

20

30

40

50

のあった全てのVSFのコンピューティング要素にこれらを自動的に分散およびインストールすることができる。

#### 【0145】

##### 実施機構

コンピューティング要素および制御ブレンは幾つかの形式で実施されてもよく、且つ本発明は特定の形式に限定されることはない。一実施例において、各コンピューティング要素は、不揮発性記憶装置1910を除き、図19に示す要素を有する汎用デジタルコンピュータであり、また制御ブレンは、上記プロセスを実施するプログラム命令の制御の下で動作する図19に示す種類の汎用デジタルコンピュータである。

#### 【0146】

図19は、本発明の実施例が実施されうるコンピュータシステム1900を示すブロック図である。コンピュータシステム1900は、情報を伝達するバス1902または他の通信機構、および情報を処理するためにバス1902に接続されたプロセッサ1904を含んでいる。コンピュータシステム1900はまた情報とプロセッサ1904が実行する命令を保存するためにバス1902に接続されたランダムアクセスメモリ(RAM)または他の動的記憶装置などのメインメモリ1906を含んでいる。メインメモリ1906も、プロセッサ1904が実行する命令の実行中に、一時的数値変数や他の中間情報を保存するのに使用することができる。コンピュータシステム1900は更に、静的情報およびプロセッサ1904の命令を保存するために、バス1902に接続されたリードオンリメモリ(ROM)1908や他の静的記憶装置を含んでいる。磁気ディスクや光ディスクなどの記憶装置1910が設けられ、情報および命令を保存するためにバス1902に接続されている。

#### 【0147】

コンピュータシステム1900は、情報をコンピュータユーザに表示するために、陰極線管(CRT)などのディスプレイ1912にバス1902を介して接続されていてもよい。英数字および他のキーを含む入力機器1914は、情報および命令の選択をプロセッサ1904に伝達するために、バス1902に接続されている。他の種類のユーザ入力機器は、方向情報および命令の選択をプロセッサ1904に伝達し、且つカーソルの動きをディスプレイ1912上で制御するためのマウス、トラックボール、カーソル方向キーなどのカーソルコントロール1916である。この入力機器は一般に、機器が平面における位置を指定することを可能にする2つの軸、すなわち第1軸(例えばx)および第2軸(例えばy)における2つの自由度を有している。

#### 【0148】

本発明は、拡張可能コンピューティングシステムを制御するための、コンピュータシステム1900の使用に関連している。本発明の一実施例によれば、拡張可能コンピューティングシステムの制御は、メインメモリ1906に含まれる一又は二以上の命令の一又は二以上のシーケンスを実行するプロセッサ1904に応じて、コンピュータシステム1900によって行われる。このような命令は、記憶装置1910などの別のコンピュータで読み取り可能な媒体からメインメモリ1906に読み込まれる。メインメモリ1906に含まれる命令のシーケンスを実行することにより、プロセッサ1904は、上記のプロセス工程を実行する。マルチ処理構成において一又は二以上のプロセッサを使用し、メインメモリ1906に含まれる命令のシーケンスを実行してもよい。別の実施例においては、配線接続された回路を、ソフトウェア命令の代わりに、あるいはこれと組み合わせて使用し、本発明を実施してもよい。従って、本発明の実施例は、ハードウェア回路およびソフトウェアの特定の組合せに限定されない。

#### 【0149】

ここで使用する用語「コンピュータで読み取り可能な媒体」は、プロセッサ1904に命令を与えて実行することに関連する媒体を意味する。このような媒体は、不揮発性媒体、揮発性媒体および伝送媒体を含むがこれらに限定されない多くの形式を取ることができる。不揮発性媒体は例えば、記憶装置1910などの光または磁気ディスクを含む。揮発性

10

20

30

40

50

媒体は、メインメモリ1906などの動的メモリを含む。伝送媒体は、バス1902を構成する配線を含む同軸ケーブル、銅線および光ファイバーを含む。伝送媒体も、無線および赤外線データ通信の間に生成されるような音波や光波の形式を取ることができる。

【0150】

コンピュータで読み取り可能な媒体の一般的な形式は、例えば、以下に説明するようなフロッピー（登録商標）ディスク、フレキシブルディスク、ハードディスク、磁気テープ、ほかの磁気媒体、CD-ROM、他の光媒体、パンチカード、紙テープ、穴のパターンを有する他の物理的媒体、RAM、PROM、EPROM、FLASH-EPROM、他のメモリチップまたはカートリッジ、搬送波、またはコンピュータが読み取り可能なほかの媒体を含む。

【0151】

コンピュータが読み取り可能な媒体の様々な形式は、プロセッサ1904に一又は二以上の命令の一又は二以上のシーケンスを送って実行させることに関連していてもよい。例えば、命令はまず、遠隔コンピュータの磁気ディスクに送られる。遠隔コンピュータはその動的メモリに命令をロードして、モデムを使用して電話回線上で命令を送る。コンピュータシステム1900に対して遠隔にあるモデムは、電話回線上のデータを受信し、赤外線トランスミッタを使用してデータを赤外線信号に変換することができる。バス1902に接続された赤外線ディテクタは、赤外線信号で運ばれるデータを受信して、バス1902にデータを出す。バス1902はデータをメインメモリ1906に送り、ここからプロセッサ1904は命令の検索と実行を行う。メインメモリ1906が受信した命令は、プロセッサ1904の実行の前または後で記憶装置1910に随意に保存することができる。

【0152】

コンピュータシステム1900は、バス1902に接続された通信インタフェース1918も含んでいる。通信インタフェース1918は、ローカルネットワーク1922に接続されたネットワークリンク1920へ接続する双方向データ通信を行う。例えば、通信インタフェース1918は、対応する種類の電話回線へのデータ通信接続を行うためのデジタル総合サービスネットワーク（ISDN）カードまたはモデムであってもよい。他の例としては、通信インタフェース1918は、互換性のあるLANへのデータ通信接続を行うためのローカルエリアネットワーク（LAN）であってもよい。無線リンクも実施することができる。このような実施において、通信インタフェース1918は、様々な種類の情報を表すデジタルデータストリームを伝える電気、電磁または光信号を送受信する。

【0153】

ネットワークリンク1920は一般に、一又は二以上のネットワークを介して、他のデータ機器へのデータ通信を行う。例えば、ネットワークリンク1920は、ローカルネットワーク1922を介して、インターネットサービスプロバイダ（ISP）1926によって運営されるホストコンピュータ1924またはデータ機器への接続を提供する。ISP1926は、一般に「インターネット」と現在呼ばれている世界規模パケットデータ通信ネットワーク1928を介してデータ通信サービスを提供する。ローカルネットワーク1922およびインターネット1928は共に、デジタルデータストリームを伝える電気、電磁または光信号を使用する。様々なネットワークおよびネットワークリンク1920上の信号、および通信インタフェース1918を介して、コンピュータシステム1900に対してデジタルデータを送受する信号は、情報を運ぶ搬送波の典型的な形である。

【0154】

コンピュータシステム1900は、ネットワーク、ネットワークリンク1920および通信インタフェース1918を介して、メッセージを送信し、且つプログラムコードを含むデータを受信することができる。インターネットの例では、サーバ1930は、インターネット1928、ISP1926、ローカルネットワーク1922、および通信インタフェース1918を介して、アプリケーションプログラムの要求コードを送信する。本発明によれば、このようなダウンロードしたアプリケーションは、ここに説明する拡張可能コンピューティングシステムの制御を規定する。

10

20

30

40

50

## 【0155】

受信コードは、受信されるとプロセッサ1904により実行、および／または後で実行するために記憶装置1910あるいは他の不揮発性ストレージに保存しておいてもよい。このように、コンピュータシステム1900は、搬送波という形でアプリケーションコードを得ることができる。

## 【0156】

ここに開示したコンピューティンググリッドは、時にパワーグリッドと呼ばれる公共電力ネットワークと概念的に比較される。パワーグリッドは、単一の大規模電力インフラストラクチャを介して電力サービスを得るために、多数の関係者に拡張可能手段を提供する。同様に、ここに開示したコンピューティンググリッドは、単一の大規模コンピューティングインフラストラクチャを使用することによって、多数の組織にコンピューティングサービスを提供する。パワーグリッドを使用するので、電力消費者はその個人電力設備を自主的に管理することはない。例えば、ユーティリティ消費者がその設備または共有設備において個人用発電機を運転させ、個人でその容量および増加を管理する理由はない。その代わりに、パワーグリッドは人口の大部分へ広範囲に電力を供給することができるので、大きなスケールメリットが得られる。同様に、ここに開示するコンピューティンググリッドは、単一の大規模なコンピューティングインフラストラクチャを使用して、人口の大部分にコンピューティングサービスを提供することができる。

## 【0157】

上記の詳述において、具体的な実施例に関連して本発明を説明した。しかしながら、本発明の広大な精神および範囲から逸脱することなく、様々な改良および変更を本発明に加えることが可能であることは明白となろう。従って、説明および図面は、限定的意味ではなく例証において考慮される。

## 【図面の簡単な説明】

## 【図1A】

図1Aは、単一のコンピューティング要素トポロジーを使用する単純なウェブサイトのブロック図である。

## 【図1B】

図1Bは、1層ウェブサーバファームのブロック図である。

## 【図1C】

図1Cは、3層ウェブサーバファームのブロック図である。

## 【図2】

図2は、ローカルコンピューティンググリッドを含む拡張可能コンピューティングシステム200の1つの構成を示すブロック図である。

## 【図3】

図3は、SANゾーンを特徴付ける典型的な仮想サーバファームのブロック図である。

## 【図4A】

図4Aは、コンピューティング要素の追加および仮想サーバファームからの要素の除去に関連する連続工程を示すブロック図である。

## 【図4B】

図4Bは、コンピューティング要素の追加および仮想サーバファームからの要素の除去に関連する連続工程を示すブロック図である。

## 【図4C】

図4Cは、コンピューティング要素の追加および仮想サーバファームからの要素の除去に関連する連続工程を示すブロック図である。

## 【図4D】

図4Dは、コンピューティング要素の追加および仮想サーバファームからの要素の除去に関連する連続工程を示すブロック図である。

## 【図5】

図5は、仮想サーバファームシステム、コンピューティンググリッド、監視機構の実施例

10

20

30

40

50

のブロック図である。

【図 6】

図 6 は、仮想サーバファームの論理接続のブロック図である。

【図 7】

図 7 は、仮想サーバファームの論理接続のブロック図である。

【図 8】

図 8 は、仮想サーバファームの論理接続のブロック図である。

【図 9】

図 9 は、制御プレーンおよびデータプレーンの論理関係のブロック図である。

【図 10】

図 10 は、マスター制御選択プロセスの状態図である。

【図 11】

図 11 は、スレーブ制御プロセスの状態図である。

【図 12】

図 12 は、マスター制御プロセスの状態図である。

【図 13】

図 13 は、中央制御プロセッサおよび多数の制御プレーンおよびコンピューティンググリッドのブロック図である。

【図 14】

図 14 は、制御プレーンおよびコンピューティンググリッドの部分を実施するアーキテクチャのブロック図である。

【図 15】

図 15 は、ファイアウォールによって保護されるコンピューティンググリッドを有するシステムのブロック図である。

【図 16】

図 16 は、制御プレーンをコンピューティンググリッドに接続するアーキテクチャのブロック図である。

【図 17】

図 17 は、VLAN タグと IP アドレスを密に結合する配置のブロック図である。

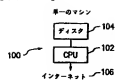
【図 18】

図 18 は、WAN 接続上で拡張した複数の VSF のブロック図である。

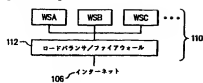
【図 19】

図 19 は、実施例が実施されるコンピュータシステムのブロック図である。

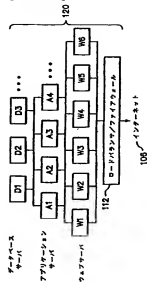
【図 1 A】



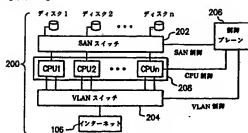
【図 1 B】



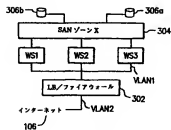
【図 1 C】



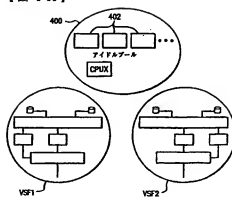
【図 2】



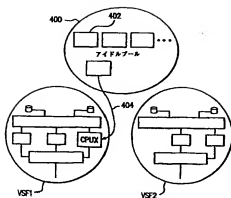
【図 3】



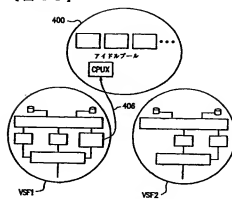
【図 4 A】



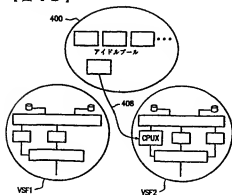
【図 4 B】



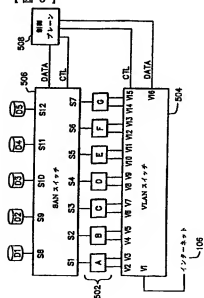
【図 4 C】



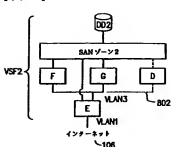
【図 4 D】



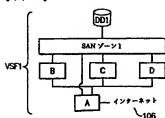
【図 5】



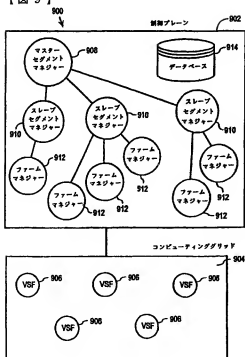
【图 8】



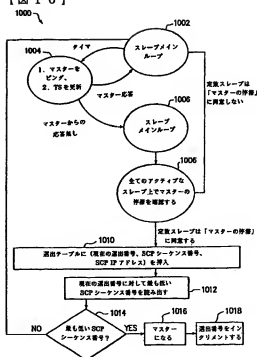
【圖 7】



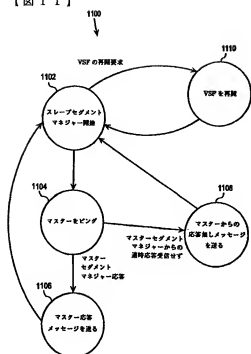
【图 9】



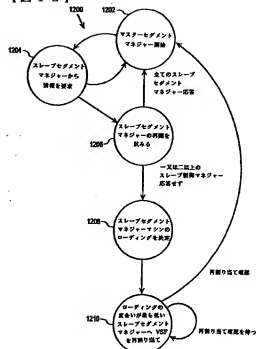
【 10 】



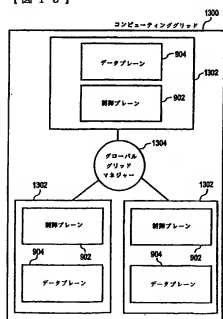
【図 11】



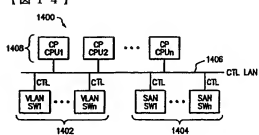
【図 12】



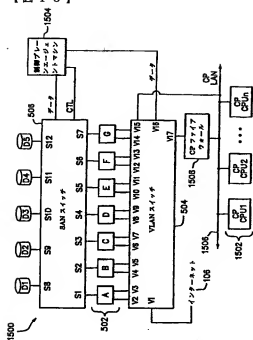
【図 13】



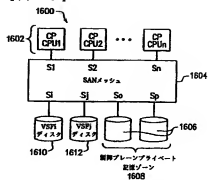
【図 14】



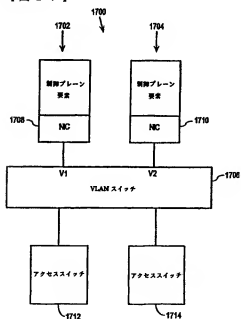
【図 15】



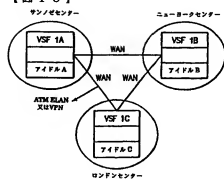
【図 16】



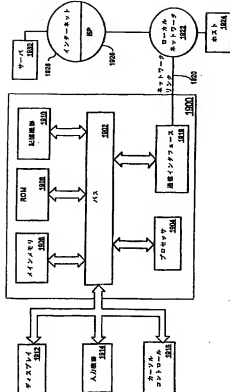
【図 17】



【図 18】



【図 19】



## WO 02/03203 A2

(19) World Intellectual Property Organization  
International Bureau



(16) International Publication Number  
WO 02/03203 A2

(43) International Publication Date  
18 January 2002 (18.01.2002)

PCT

(53) International Patent Classification?

DATA FILE

(21) International Application Number: PCT/JP2011/0653

(22) International Filing Date: 13 June 2001 (13.06.2001)

(2) Filing Language:	English
----------------------	---------

(24) Publication Language: English

(38) Priority Dates:		
03/21/2000	30 June 2000 (30.06.2000)	US
03/03/2000	2 August 2000 (02.08.2000)	US

(7) Applicant TERRASPRING, INC. CLAIMS 4900142.

(17) Investors: ARIZ, Asher; 4180 Theragey Common, Fremont, CA 94533 (US); MARKSON, Peter; 30 Monte Real, San Mateo, CA 94402 (US); PATTERSON, Martin; 1445 Merry Street, Mountain View, CA 94041 (US); GRAY, Mark; 664 Palmrose Avenue, Mountain View, CA 94041 (US).

(74) Agents: **BECKER, Edward et al**; Hickman Petersen  
Dwyer & Becker, LLP, 1600 Wilcox Street, San Jose, CA  
95125 (UT).

(31) Designated States (according to AIX, A2, A3, A4, A5, A6, A7, A8, A9, B1, B2, B3, B4, B5, B6, B7, B8, B9, C1, C2, C3, C4, C5, C6, C7, C8, C9, D1, D2, D3, D4, D5, D6, D7, D8, D9, E1, E2, E3, E4, E5, E6, E7, E8, E9, F1, F2, F3, F4, F5, F6, F7, F8, F9, G1, G2, G3, G4, G5, G6, G7, G8, G9, H1, H2, H3, H4, H5, H6, H7, H8, H9, I1, I2, I3, I4, I5, I6, I7, I8, I9, J1, J2, J3, J4, J5, J6, J7, J8, J9, K1, K2, K3, K4, K5, K6, K7, K8, K9, L1, L2, L3, L4, L5, L6, L7, L8, L9, M1, M2, M3, M4, M5, M6, M7, M8, M9, N1, N2, N3, N4, N5, N6, N7, N8, N9, O1, O2, O3, O4, O5, O6, O7, O8, O9, P1, P2, P3, P4, P5, P6, P7, P8, P9, Q1, Q2, Q3, Q4, Q5, Q6, Q7, Q8, Q9, R1, R2, R3, R4, R5, R6, R7, R8, R9, S1, S2, S3, S4, S5, S6, S7, S8, S9, T1, T2, T3, T4, T5, T6, T7, T8, T9, U1, U2, U3, U4, U5, U6, U7, U8, U9, V1, V2, V3, V4, V5, V6, V7, V8, V9, W1, W2, W3, W4, W5, W6, W7, W8, W9, X1, X2, X3, X4, X5, X6, X7, X8, X9, Y1, Y2, Y3, Y4, Y5, Y6, Y7, Y8, Y9, Z1, Z2, Z3, Z4, Z5, Z6, Z7, Z8, Z9).

(B4) Designated States (regions): ARIPO patent (BM, CH, DE, ES, FR, GB, GR, IE, IT, NL, PT, SE, SI, SK, CZ, PL, TR, UK, JP), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FR, GB, GR, IE, IT, LI, LU, MC, NL, PT, SE, TH), OAPI patent (BF, BI, CF, CI, CM, CG, CO, CR, CU, EE, EG, GN, GP, HT, IL, IN, KE, MG, ML, MR, MU, NE, NG, NO, NY, OM, PG, PH, RO, SD, SG, SN, ST, SV, SZ, TD, TG, TN, TZ, UA, UG, UZ, ZA, ZM, ZW).

- without international search report and to be republished upon receipt of that report
- solely in electronic form (except for this from page) and available upon request from the International Bureau

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(16) Title: METHOD AND APPARATUS FOR CONTROLLING AN EXTENSIBLE COMPUTING SYSTEM

(37) **Abstract:** Methods and apparatus providing, controlling and managing a dynamically sized, highly available and available server farm are disclosed. A Virtual Server Farm (VSF) is created out of a wide range computing hosts ("Computing Ods") which is physically concentrated onto and then logically divided into VSEs for various organizations on demand. Each organization serves various types of applications, and each VSE is dynamically sized within the Computing Ods, allocation and control of the elements in the VSE is performed in a VSE, a VSE is dynamically sized to all applications and storage elements in the VSE, and the VSE is dynamically sized to all applications and storage elements in the VSE. The physical topology of the VSE is created out of the network and the physical topology is necessary in order to connect VSEs in many different configurations, including single-line Web server or multi-line Web server, application server, database server configurations.

WO 02/02203

PCT/US96/01953

METHOD AND APPARATUS FOR CONTROLLING AN  
EXTENSIBLE COMPUTING SYSTEM

FIELD OF THE INVENTION

The present invention relates generally to data processing. The invention relates more specifically to a method and apparatus for controlling a computing grid.

BACKGROUND OF THE INVENTION

Builders of Web sites and other computer systems today are faced with many challenging systems planning issues. These issues include capacity planning, site availability and site security. Accomplishing these objectives requires finding and hiring trained personnel capable of engineering and operating a site, which may be potentially large and complicated. This has proven to be difficult for many organizations because designing, constructing and operating large sites is often outside their core business.

One approach has been to host an enterprise Web site at a third party site, co-located with other Web sites of other enterprises. Such outsourcing facilities are currently available from companies such as Rixdata, AboveNet, GlobalCenter, etc. These facilities provide physical space and redundant network and power facilities shared by multiple customers.

Although outsourcing web site hosting greatly reduces the task of establishing and maintaining a web site, it does not relieve a company of all of the problems associated with maintaining a web site. Companies must still perform many tasks relating to their computing infrastructures in the course of building, operating and growing their facilities. Information technology managers of the enterprises hosted at such facilities remain responsible for manually selecting, installing, configuring, and maintaining their own computing equipment at the facilities. The managers must still confront difficult issues such as resource planning and handling peak capacity. Specifically, managers must estimate resource demands and request resources from the outsourcing company to handle the demands. Many managers ensure sufficient capacity by requesting substantially more resources than are needed to provide a cushion against unexpected peak demands. Unfortunately, this often results in significant amounts of unused capacity that increases companies' overhead for hosting their web sites.

Even when outsourcing companies also provide complete computing facilities including servers, software and power facilities, the facilities are no easier to scale and

WO 02/03202

PCT/US00/19653

grow for the outsourcing company, because growth involves the same general and expensive administrative steps. In addition, problems remain with capacity planning for unexpected peak demand. In this situation, the outsourcing companies often maintain significant amounts of unused capacity.

Further, Web sites managed by outsourcing companies often have different requirements. For example, some companies may require the ability to independently administer and control their Web sites. Other companies may require a particular type or level of security that isolates their Web sites from all other sites that are co-located at an outsourcing company. As another example, some companies may require a secure connection to an enterprise Intranet located elsewhere.

Also, various Web sites differ in internal topology. Some sites simply comprise a row of Web servers that are load balanced by a Web load balancer. Some load balancers are Load Director from Cisco Systems, Inc., Nginx from F5 Networks, Web Director from Altiris, etc. Other sites may be constructed in a multi-tier fashion, whereby a row of Web servers handles Hypertext Transfer Protocol (HTTP) requests, but the bulk of the application logic is implemented in separate application servers. These application servers in turn may need to be connected back to a tier of database servers.

Some of these different configuration scenarios are shown in FIG. 1A, FIG. 1B, and FIG. 1C. FIG. 1A is a block diagram of a simple Web site, comprising a single computing element or machine 100 that includes a CPU 102 and disk 104. Machine 100 is coupled to the global, packet-switched data network, known as the Internet 106, or to another network. Machine 100 may be housed in a co-location service of the type described above.

FIG. 1B is a block diagram of a 1-tier Web server farm 110 comprising a plurality of Web servers WSA, WSB, WSC. Each of the Web servers is coupled to a load balancer 112 that is coupled to Internet 106. The load balancer divides the traffic between the servers to maintain a balanced processing load on each server. Load balancer 112 may also include or may be coupled to a firewall for protecting the Web servers from unauthorized traffic.

FIG. 1C shows a 3-tier server farm 120 comprising a tier of Web servers W1, W2, etc., a tier of application servers A1, A2, etc., and a tier of database servers D1, D2, etc. The Web servers are provided for handling HTTP requests. The application servers execute the bulk of the application logic. The database servers execute database management system (DBMS) software.

WO 2003/02323

PCT/JP03/00653

Given the diversity in topology of the kinds of Web sites that need to be constructed and the varying requirements of the corresponding companies, it may appear that the only way to construct large-scale Web sites is to physically construct build each site. Indeed, this is the conventional approach. Many organizations are separately struggling with the same issues, and custom building each Web site from scratch. This is inefficient and involves a significant amount of duplicate work at different enterprises.

Still another problem with the conventional approach is resource and capacity planning. A Web site may receive vastly different levels of traffic on different days or at different hours within each day. At peak traffic times, the Web site hardware or software may be unable to respond to requests in a reasonable time because it is overloaded. At other times, the Web site hardware or software may have excess capacity and be underutilized. In the conventional approach, finding a balance between having sufficient hardware and software to handle peak traffic, without incurring excessive costs or having over-capacity, is a difficult problem. Many Web sites never find the right balance and chronically suffer from under-capacity or excess capacity.

Yet another problem is failure induced by human error. A great potential hazard present in the current approach of using manually constructed server farms is that human error in configuring a new server into a live server farm can cause the server farm to malfunction, possibly resulting in loss of service to users of that Web site.

Based on the foregoing, there is a clear need in this field for improved methods and apparatuses for providing a computing system that is instantly and easily extensible on demand without requiring custom construction.

There is also a need for a computing system that supports creation of multiple segregated processing nodes, each of which can be expanded or collapsed as needed to account for changes in traffic throughput.

There is a further need for a method and apparatus for controlling such an extensible computing system and its constituent segregated processing nodes. Other needs will become apparent from the disclosure provided herein.

#### SUMMARY OF THE INVENTION

According to one aspect of the invention, the foregoing needs, and other needs and that will become apparent from the following description, are achieved by a method and apparatus for controlling and managing a highly scalable, highly available and secure data processing sites, based on a wide scale computing fabric ("computing grid"). The

WO 02/02363

PCT/US98/10653

computing grid is physically constructed once, and then logically divided up for various organizations on demand. The computing grid comprises a large plurality of computing elements that are coupled to one or more VLAN switches and to one or more storage area network (SAN) switches. A plurality of storage devices are coupled to the SAN switches and may be selectively coupled to one or more of the computing elements through appropriate switching logic and commands. One part of the VLAN switch is coupled to an external network, such as the Internet. A supervisory mechanism, layer, machine or process is coupled to the VLAN switches and SAN switches.

Initially, all storage devices and computing elements are assigned to life Pools. Under program control, the supervisory mechanism dynamically configures the VLAN switches and SAN switches to couple their ports to one or more computing elements and storage devices. As a result, each element and device are logically removed from the life Pools and become part of one or more virtual server farms (VSFs) or instant data centers (IDCs). Each VSF computing element is pointed to or otherwise associated with a storage device that contains a boot image usable by the computing element for bootstrap operation and production execution.

According to one aspect of the invention, the supervisory layer is a control plane comprised of a control mechanism hierarchy that includes one or more master control process mechanisms communicatively coupled to one or more slave control process mechanisms. The one or more master control process mechanisms allocate and deallocate slave control process mechanisms based upon slave control process mechanisms loading. The one or more master control process mechanisms instruct the slave control process mechanisms to establish IDCs by assigning subsets of processing and storage resources. The one or more master control process mechanisms perform periodic health checks on the slave control process mechanisms. Non-responsive or failed slave control mechanisms are restarted. Additional slave control mechanisms are initiated to replace slave control mechanisms that cannot be restarted. The slave control mechanisms perform periodic health checks on the master control mechanisms. When a master slave control process mechanism has failed, a slave control process mechanism is elected to be a new master control process mechanism to replace the failed master control process mechanism.

Physically constructing the computing grid once, and securely and dynamically allocating portions of the computing grid to various organizations on demand achieve economies of scale that are difficult to achieve when creating a custom build of each site.

WO 02/03203

PCT/US99/00653

## BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example, and not by way of limitation, in the figures of the accompanying drawings and in which like reference numerals refer to similar elements and in which:

FIG. 1A is a block diagram of a simple Web site having a single computing element topology.

FIG. 1B is a block diagram of a one-tier Web server farm.

FIG. 1C is a block diagram of a three-tier Web server farm.

FIG. 2 is a block diagram of one configuration of an extensible computing system 200 that includes a local computing grid.

FIG. 3 is a block diagram of an exemplary virtual server farm functioning a SAN Zone.

FIG. 4A, FIG. 4B, FIG. 4C, and FIG. 4D are block diagrams showing successive steps involved in adding a computing element and removing element from a virtual server farm.

FIG. 5 is a block diagram of an embodiment of a virtual server farm system, computing grid, and supervisory mechanisms.

FIG. 6 is a block diagram of logical connections of a virtual server farm.

FIG. 7 is a block diagram of logical connections of a virtual server farm.

FIG. 8 is a block diagram of logical connections of a virtual server farm.

FIG. 9 is a block diagram of a logical relationship between a control plane and a data plane.

FIG. 10 is a state diagram of a master control election process.

FIG. 11 is a state diagram for a slave control process.

FIG. 12 is a state diagram for a master control process.

FIG. 13 is a block diagram of a control control processor and multiple control planes and computing grids.

FIG. 14 is a block diagram of an architecture for implementing portions of a control plane and a computing grid.

FIG. 15 is a block diagram of a system with a computing grid that is protected by a firewall.

FIG. 16 is a block diagram of an architecture for connecting a control plane to a computing grid.

WO 01/03263

PC/P2004/00863

FIG. 17 is a block diagram of an arrangement for enforcing tight binding between VLAN tags and IP addresses.

FIG. 18 is a block diagram of a plurality of VSFs extended over WAN connections.

FIG. 19 is a block diagram of a computer system with which an embodiment may be implemented.

#### DETAILED DESCRIPTION OF THE INVENTION

In the following description, for the purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be apparent, however, to one skilled in the art that the present invention may be practiced without these specific details. In other instances, well-known structures and devices are shown in block diagram form in order to avoid unnecessarily obscuring the present invention.

#### VIRTUAL SERVER FARM (VSF)

According to one embodiment, a wide scale computing fabric ("computing grid") is provided. The computing grid may be physically constructed once, and then logically partitioned on demand. A part of the computing grid is allocated to each of a plurality of enterprises or organizations. Each organization's logical portion of the computing grid is referred to as a Virtual Server Farm (VSF). Each organization retains independent administrative control of its VSF. Each VSF can change dynamically in terms of number of CPUs, storage capacity and disk and network bandwidth based on real-time demands placed on the server farm or other factors. Each VSF is secure from every other organization's VSF, even though they are all logically created out of the same physical computing grid. A VSF can be connected back to an Internet using either a private leased line or a Virtual Private Network (VPN), without exposing the Internet to other organizations' VSFs.

An organization can access only the data and computing elements in the portion of the computing grid allocated to it, that is, in its VSF, even though it may exercise full (e.g. super-user or root) administrative access to these computers and can observe all traffic on Local Area Networks (LANs) to which these computers are connected. According to one embodiment, this is accomplished using a dynamic firewalling scheme, where the security perimeter of the VSF expands and shrinks dynamically. Each VSF can

WO 02/03283

FCT/00002/9653

be used to host the content and applications of an organization that may be accessed via the Internet, Intranet or Extranet.

Configuration and control of the computing elements and their associated networking and storage elements is performed by a supervisory mechanism that is not directly accessible through any of the computing elements in the computing grid. For convenience, in this document the supervisory mechanism is referred to generally as a control plane and may comprise one or more processors or a network of processors. The supervisory mechanism may comprise a Supervisor, Controller, etc. Other approaches may be used, as described herein.

The control plane is implemented on a completely independent set of computing elements assigned for supervisory purposes, such as one or more servers that may be interconnected in a network or by other means. The control plane performs control actions on the computing, networking and storage elements of the computing grid through special control ports or interfaces of the networking and storage elements in the grid. The control plane provides a physical interface to switching elements of the system, monitors loads of computing elements in the system, and provides administrative and management functions using a graphical user interface or other suitable user interface.

Computers used to implement the control plane are logically invisible to computers in the computing grid (and therefore in any specific VSF) and cannot be attacked or subverted in any way via elements in the computing grid or from external computers. Only the control plane has physical connections to the control ports on devices in the computing grid, which controls membership in a particular VSF. The devices in the computing can be configured only through these special control ports, and therefore computing elements in the computing grid are unable to change their security perimeter or access storage or computing devices which they are not authorized to do.

Thus, a VSF allows organizations to work with computing facilities that appear to comprise a private server farm, dynamically created out of a large-scale shared computing infrastructure, namely the computing grid. A control plane coupled with the computing architecture described herein provides a private server farm whose privacy and integrity is protected through access control mechanisms implemented in the hardware of the devices of the computing grid.

The control plane controls the internal topology of each VSF. The control plane can take the basic interconnection of computers, network switches and storage network switches described herein and use them to create a variety of server farm configurations.

WO 02/07283

PC/T/0001/1953

These include but are not limited to, single-tier Web server farms front-ended by a load balancer, as well as multi-tier configurations, where a Web server talks to an application server, which in turn talks to a database server. A variety of load balancing, multi-tiering and firewalling configurations are possible.

#### THE COMPUTING GRID

The computing grid may exist in a single location or may be distributed over a wide area. First this document describes the computing grid in the context of a single building-sized network, composed purely of local area technologies. Then the document describes the case where the computing grid is distributed over a wide area network (WAN).

FIG. 2 is a block diagram of one configuration of an extensible computing system 200 that includes a local computing grid 208. In this document "extensible" generally means that the system is flexible and scalable, having the capability to provide increased or decreased computing power to a particular enterprise or user upon demand. The local computing grid 208 is composed of a large number of computing elements CPU1, CPU2, ... CPUs. In an exemplary embodiment, there may be 16,000 computing elements, or more. These computing elements do not contain or store any long-lived per-element state information, and therefore may be configured without persistent or non-volatile storage such as a local disk. Instead, all long-lived state information is stored separate from the computing elements, on disks DISK1, DISK2, ... DISKs that are coupled to the computing elements via a Storage Area Network (SAN) comprising one or more SAN Switches 202. Examples of suitable SAN switches are commercially available from Brocade and Exotel.

All of the computing elements are interconnected to each other through one or more VLAN switches 204 which can be divided up into Virtual LANs (VLANs). The VLAN switches 204 are coupled to the Internet 106. In general a computing element contains one or two network interfaces connected to the VLAN switch. For the sake of simplicity, in FIG. 2 all nodes are shown with two network interfaces, although some may have less or more network interfaces. Many commercial vendors now provide switches supporting VLAN functionality. For example, suitable VLAN switches are commercially available from Cisco Systems, Inc. and 3Com Networks. Similarly there are a large number of commercially available products to construct SANs, including Fibre Channel

WO 02/02802

PCT/JP99/019655

switches, SCSI-to-Fibre-Channel bridging devices, and Network Attached Storage (NAS) devices.

Control plane 206 is coupled by a SAN Control path, CPU Control path, and VLAN Control path to SAN switches 202, CPUs CPU1, CPU2, ... CPUs, and VLAN Switches 204, respectively.

Each VSF is composed of a set of VLANs, a set of computing elements that are attached to the VLANs, and a subset of the storage available on the SAN that is coupled to the set of computing elements. The subset of the storage available on the SAN is referred to as a SAN Zone and is protected by the SAN hardware from access from computing elements that are part of other SAN zones. Preferably, VLANs that provide non-disposable port identifiers are used to prevent one customer or end user from obtaining access to VSF resources of another customer or end user.

FIG. 3 is a block diagram of an exemplary virtual server farm featuring a SAN Zone. A plurality of Web servers WS1, WS2, etc., are coupled by a first VLAN (VLAN1) to a load balancer (LB)/firewall 302. A second VLAN (VLAN2) couples the Internet 106 to the load balancer (LB)/firewall 302. Each of the Web servers may be selected from among CPU1, CPU2, etc., using mechanisms described further herein. The Web servers are coupled to a SAN Zone 304, which is coupled to one or more storage devices 306a, 306b.

At any given point in time, a computing element in the computing grid, such as CPU1 of FIG. 2, is only connected to the set of VLANs and the SAN zone(s) associated with a single VSF. A VSF typically is not shared among different organizations. The subset of storage on the SAN that belongs to a single SAN zone, and the set of VLANs associated with it and the computing elements on these VLANs define a VSF.

By controlling the membership of a VLAN and the membership of a SAN zone, control plane enforces a logical partitioning of the computing grid into multiple VSFs. Members of one VSF cannot access the computing or storage resources of another VSF. Such access restrictions are enforced at the hardware level by the VLAN switches, and by port-level access control mechanisms (e.g., zoning) of SAN hardware such as Fibre Channel switches and edge devices such as SCSI to Fibre Channel bridging hardware. Computing elements that form part of the computing grid are not physically connected to the control plane or interfaces of the VLAN switches and the SAN switches, and therefore cannot control the membership of the VLANs or SAN zones. Accordingly, the

WO 01/62263

PCP/EDU/10653

computing elements of the computing grid cannot access computing elements not located in the VSF in which they are contained.

Only the computing elements that are the control plane are physically connected to the control ports or interfaces of the devices in the grid. Devices in the computing grid (computers, SAN switches and VLAN switches) can only be configured through such control ports or interfaces. This provides a simple yet highly secure means of enforcing the dynamic partitioning of the computing grid into multiple VSFs.

Each computing element in a VSF is replaceable by any other computing element. The number of computing elements, VLANs and SAN ports associated with a given VSF may change over time under control of the control plane.

In one embodiment, the computing grid includes an Mite Pool that comprises large number of computing elements that are kept in reserve. Computing elements from the Mite Pool may be assigned to a particular VSF for reasons such as increasing the CPU or memory capacity available to that VSF, or to deal with failure of a particular computing element in a VSF. When the computing elements are configured as Web servers, the Mite Pool serves as a large "shock absorber" for varying or "bursty" Web traffic loads and related peak processing loads.

The Mite Pool is shared between many different organizations, and therefore it provides economies of scale, since no single organization has to pay for the entire cost of the Mite Pool. Different organizations can obtain computing elements from the Mite Pool at different times in the day, as needed, thereby enabling each VSF to grow when required and shrink when traffic falls down to normal. If many different organizations continue to peak at the same time and thereby potentially exhaust the capacity of the Mite Pool, the Mite Pool can be increased by adding more CPUs and storage elements to it (scalability). The capacity of the Mite Pool is engineered so as to greatly reduce the probability that, in steady state, a particular VSF may not be able to obtain an additional computing element from the Mite Pool when it needs to.

FIG. 4A, FIG. 4B, FIG. 4C, and FIG. 4D are block diagrams showing successive steps involved in moving a computing element in and out of the Mite Pool. Referring first to FIG. 4A, assume that the control plane has logically connected elements of the computing grid into first and second VSFs labeled VSF1, VSF2. Mite Pool 400 comprises a plurality of CPUs 402, one of which is labeled CPUX. In FIG. 4B, VSF1 has developed a need for an additional computing element. Accordingly, the control plane moves CPUX from Mite Pool 400 to VSF1, as indicated by path 404.

WO 02/03203

PCT/US01/09653

In FIG. 4C, VSF1 no longer needs CPUX, and therefore the control plane moves CPUX out of VSF1 and back into the Idle Pool 400. In FIG. 4D, VSF2 has developed a need for an additional computing element. Accordingly, the control plane moves CPUX from the Idle Pool 400 to VSF2. Thus, over the course of time, as traffic conditions change, a single computing element may belong to the Idle Pool (FIG. 4A), then be assigned to a particular VSF (FIG. 4B), then be placed back in the Idle Pool (FIG. 4C), and then belong to another VSF (FIG. 4D).

At each one of these stages, the control plane configures the LAN switches and SAN switches associated with that computing element to be part of the VLANs and SAN zones associated with a particular VSF (or the Idle Pool). According to one embodiment, in between each transition, the computing element is powered down or rebooted. When the computing element is powered back up, the computing element views a different portion of storage zone on the SAN. In particular, the computing element views a portion of storage zone on the SAN that includes a bootable image of an operating system (e.g., Linux, NT, Solaris, etc.). The storage zone also includes a data portion that is specific to each organization (e.g., files associated with a Web server, database partitions, etc.). The computing element is also part of another VLAN which is part of the VLAN set of another VSF, so it can access CPUs, SAN storage devices and NAS devices associated with the VLANs of the VSF into which it has been transitioned.

In a preferred embodiment, the storage zones include a plurality of pre-defined logical blueprints that are associated with roles that may be assumed by the computing elements. Initially, no computing element is dedicated to any particular role or task such as Web server, application server, database server, etc. The role of the computing element is acquired from one of a plurality of pre-defined, stored blueprints, each of which defines a boot image for the computing elements that are associated with that role. The blueprints may be stored in the form of a file, a database table, or any other storage format that can associate a boot image location with a role.

Thus, the movements of CPUX in FIG. 4A, FIG. 4B, FIG. 4C, FIG. 4D are logical, not physical, and are accomplished by re-configuring VLAN switches and SAN Zones under control of the control plane. Further, each computing element in the computing grid initially is essentially fungible, and assumes a specific pre-defined role only after it is connected in a virtual server farm and loads software from a boot image. No computing element is dedicated to any particular role or task such as Web server, application server, database server, etc. The role of the computing element is acquired

W0 01/01263

PCT/US00/010963

from one of a plurality of pre-defined, stored blueprints, each of which is associated with a role, each of which defines a host image for the computing elements that are associated with that role.

Since there is no long-lived state information stored in any given computing element (such as a local disk), nodes are easily moved between different VSPs, and can run completely different OS and application software. This also makes each computing element highly replaceable, in case of planned or unplanned downtime.

A particular computing element may perform different roles as it is brought into and out of various VSPs. For example, a computing element may act as a Web server in one VSP, and when it is brought into a different VSP, it may be a database server, a Web load balancer, a Firewall, etc. It may also successively boot and run different operating systems such as Linux, NT or Solaris in different VSPs. Thus, each computing element in the computing grid is flexible, and has no static role assigned to it. Accordingly, the entire reserve capacity of the computing grid can be used to provide any of the services required by any VSP. This provides a high degree of availability and reliability to the services provided by a single VSP, because each server performing a particular service has potentially thousands of back-up servers able to provide the same service.

Further, the large reserve capacity of the computing grid can provide both dynamic load balancing properties, as well as high processor availability. This capability is enabled by the unique combination of identical computing elements interconnected via VLANs, and connected to a configurable mass of storage devices via a SAN, all controlled in real-time by the control plane. Every computing element can act in the role of any required server in any VSP, and can connect to any logical partition of any disk in the SAN. When the grid requires more computing power or disk capacity, computing elements or disk storage is manually added to the idle pool, which may decrease over time as more organizations are provided VSP services. No manual intervention is required in order to increase the number of CPUs, network and disk bandwidth and storage available to a VSP. All such resources are allocated on demand from CPU, network and disk resources available in the Idle Pool by the control plane.

A particular VSP is not subjected to manual reconfiguration. Only the computing elements in the idle pool are manually configured into the computing grid. As a result, a great potential hazard present in current manually constructed server farms is removed. The possibility that human error in configuring a new server into a live server farm can

WO 02/02003

PCT/US00/09653

causes the server farm to malfunction, possibly resulting in loss of service to users of that Web site, is virtually eliminated.

The control plane also replicates data stored in SAN attached storage devices, so that failure of any particular storage element does not cause a loss of service to any part of the system. By decoupling long-lived storage from computing devices using SANs, and by providing redundant storage and computing elements, where any computing element can be attached to any storage partition, a high degree of availability is achieved.

#### A DETAILED EXAMPLE OF ESTABLISHING A VIRTUAL SERVER FARM, ADDING A PROCESSOR TO IT, AND REMOVING A PROCESSOR FROM IT

FIG. 5 is a block diagram of a computing grid and control plane mechanism according to an embodiment. With reference to FIG. 5, the following describes the detailed steps that may be used to create a VSF, add nodes to it and delete nodes from it.

FIG. 5 depicts computing elements 502, computing computers A through G, coupled to VLAN capable switch 504. VLAN switch 504 is coupled to Internet 106, and the VLAN switch has ports V1, V2, etc. Computers A through G are further coupled to SAN switch 506, which is coupled to a plurality of storage devices or disks D1-D5. The SAN switch 506 has ports S1, S2, etc. A control plane mechanism 508 is communicatively coupled by control paths and data paths to SAN switch 506 and to VLAN switch 504. The control plane is able to send control commands to those devices through the control paths.

For the sake of simplicity and exposition, the number of computing elements in FIG. 5 is a small number. In practice, a large number of computers, e.g., thousands or more, and an equally large number of storage devices form the computing grid. In such larger structures, multiple SAN switches are interconnected to form a mesh, and multiple VLAN switches are interconnected to form a VLAN mesh. For clarity and simplicity, however, FIG. 5 shows a single SAN switch and a single VLAN switch.

Initially, all computers A-G are assigned to the idle pool until the control plane receives a request to create a VSF. All ports of the VLAN switch are assigned to a specific VLAN which we shall label as VLAN 1 (for the idle zone). Assume that the control plane is asked to construct a VSF, containing one load balancer/firewall and two Web servers connected to a storage device on the SAN. Requests to control plane may arrive through a management interface or other computing element.

WO 03/01203

PCT/US99/09653

In response, the control plane assigns or allocates CPU A as the load balancer/firewall, and allocates CPUs B and C as the Web servers. CPU A is logically placed in SAN Zone 1, and pointed to a bootable partition on a disk that contains dedicated load balancing/firewalling software. The term "pointed to" is used for convenience and is intended to indicate that CPU A is given, by any means, information sufficient to enable CPU A to obtain or locate appropriate software that it needs to operate. Placement of CPU A in SAN Zone 1 enables CPU A to obtain resources from disks that are controlled by the SAN of that SAN Zone.

The load balancer is configured by the control plane to know about CPUs B and C as the two Web servers it is supposed to load balance. The firewall configuration protects CPUs B and C against unauthorized access from the Internet 196. CPUs B and C are pointed to a disk partition on the SAN that contains a bootable OS image for a particular operating system (e.g., Solaris, Linux, NT etc) and Web server application software (e.g., Apache). The VLAN switch is configured to place ports v1 and v2 on VLAN 1, and ports v3, v4, v5, v6 and v7 on VLAN 2. The control plane configures the SAN switch 506 to place Fibre-Channel switch ports s1, s2, s3 and s4 into SAN zone 1.

A description of how a CPU is pointed to a particular disk drive, and what this means for booting up and shared access to disk data, is provided further herein.

FIG. 6 is a block diagram of the resulting logical connectivity of computing elements, which are collectively called VSF 1. Disk drive DD1 is selected from among storage devices D1, D2, etc. Once the logical structure as shown in FIG. 6 is achieved, CPUs A, B, C are given a power-up command. In response, CPU A becomes a dedicated load balancer/firewall-computing element, and CPUs B, C become Web servers.

Now, assume that because of a policy-based rule, the control plane determines that another Web server is required in VSF 1. This may be caused, for example, by an increased number of requests to the Web site and the customer's plan permits at least three Web servers to be added to VSF 1. Or it may be because the organization that owns or operates the VSF wants another server, and has added it through an administrative mechanism, such as a privileged Web page that allows it to add more servers to its VSF.

In response, the control plane decides to add CPU D to VSF 1. In order to do this, the control plane will add CPU D to VLAN 2 by adding ports v8 and v9 to VLAN 2. Also, CPU D's SAN port s4 is added to SAN zone 1. CPU D is pointed to a bootable portion of the SAN storage that boots up and runs as a Web server. CPU D also gets read-only access to the shared data on the SAN, which may consist of Web page contents,

WO 03/03203

PCT/JP99/00653

executable server scripts, etc. This way it is able to serve Web requests intended for the server farm such as CPU's B and C serve requests. The control plane will also configure the load balancer (CPU A) to include CPU D as part of the server set which is being load balanced.

CPU D is now booted up, and the size of the VSF has now increased to three Web servers and 1 load balancer. FIG. 7 is a block diagram of the resulting logical connectivity.

Assume that the control plane now receives a request to create another VSF, which it will name VSF 2, and which needs two Web servers and one load balancer/firewall. The control plane allocates CPU B to be the load balancer/firewall and CPU's F, G to be the Web servers. It configures CPU B to know about CPU's F, G as the two computing elements to load balance against.

To implement this configuration, the control plane will configure VLAN switch 504 to include port v10, v11 in VLAN 1 (that is, connected to the Internet 106) and ports v12, v13 and v14, v15 to be in VLAN 3. Similarly, it configures SAN switch 506 to include SAN ports s6 and s7 and s9 in SAN zone 2. This SAN zone includes the storage containing the software necessary to run CPU B as a load-balancer and CPU's F and G as Web servers that use a shared read-only disk partition contained in Disk D2 in SAN zone 2.

FIG. 8 is a block diagram of the resulting logical connectivity. Although two VSFs (VSF 1, VSF 2) share the same physical VLAN switch and SAN switch, the two VSFs are logically partitioned. Users who access CPU's B, C, D, or the enterprise that owns or operates VSF 1 can only access the CPU's and storage of VSF 1. Such users cannot access the CPU's or storage of VSF 2. This occurs because of the combination of the separate VLANs and the 2 firewalls on the only shared segment (VLAN 1), and the different SAN zones in which the two VSFs are configured.

Further assume that later, the control plane decides that VSF 1 can now fall back down to two Web servers. This may be because the temporary increase in load on VSF 1 has decreased, or it may be because of some other administrative action taken. In response, the control plane will shut down CPU D by a special command that may include powering down the CPU. Once the CPU has shut down, the control plane removes ports v6 and v9 from VLAN 2, and also removes SAN port s6 from SAN zone 1. Port s6 is placed in an idle SAN zone. The idle SAN zone may be designated, for example, SAN Zone 1 (for Idle) or Zone 0.

WO 02/02303

PCT/JP00/00553

Some time later, the control plane may decide to add another node to VSF 2. This may be because the load on the Web servers in VSF 2 has temporarily increased or it may be due to other reasons. Accordingly, the control plane decides to place CPU D in VSF 2, as indicated by dashed path 402. In order to do this, it configures the VLAN switch to include ports v8, v9 in VLAN 3 and SAN port 4 in SAN zone 2. CPU D is pointed to the portion of the storage on disk device 2 that contains a bootable image of the OS and Web server software required for servers in VSF 2. Also, CPU D is granted read-only access to data in a file system shared by the other Web servers in VSF 2. CPU D is powered back up, and it now runs as a load-balanced Web server in VSF 2, and can no longer access any data in SAN zone 1 or the CPUs attached to VLAN 2. In particular, CPU D has no way of accessing any element of VSF 1, even though at an earlier point in time it was part of VSF 1.

Further, in this configuration, the security perimeter enforced by CPU B has dynamically expanded to include CPU D. Thus, embodiments provide dynamic firewalling that automatically adjusts to properly protect computing elements that are added to or removed from a VSF.

For purposes of explanation, embodiments have been described herein in the context of port-based SAN zoning. Other types of SAN zoning may also be used. For example, LUN level SAN zoning may be used to create SAN zones based upon logical volumes within disk arrays. An example product that is suitable for LUN level SAN zoning is the Volume Logic Product from BMC Corporation.

#### DISK DEVICES ON THE SAN

There are several ways by which a CPU can be pointed to a particular device on the SAN, for booting up purposes, or for accessing disk storage which needs to be shared with other nodes, or otherwise provided with information about where to find bootstrap programs and data.

One way is to provide a SCSI-to-Fibre Channel bridging device attached to a computing element and a SCSI interface for the local disks. By routing that SCSI port to the right drive on the Fibre-Channel SAN, the computer can access the storage device on the Fibre-Channel SAN just as it would access a locally attached SCSI disk. Therefore, software such as boot-up software simply boots off the disk device on the SAN just as it would boot off a locally attached SCSI disk.

WO 02/0281

PC/FIBER/2003

Another way is to have a Fibre-Channel interface on the node and associated device-driver and boot ROM and OS software that permits the Fibre-Channel interface to be used as a boot device.

Yet another way is to have an interface card (e.g., PCI bus or Smart) which appears to be a SCSI or IDE device controller but that in turn communicates over the SAN to access the disk. Operating systems such as Solaris integrally provide diskless boot functions that can be used in this alternative.

Typically there will be two kinds of SAN disk devices associated with a given node. The first is one which is not logically shared with other computing elements, and constitutes what is normally a per-node root partition containing bootable OS images, local configuration files, etc. This is the equivalent of the root file system on a Unix system.

The second kind of disk is shared storage with other nodes. The kind of sharing varies by the OS software running on the CPU and the needs of the nodes accessing the shared storage. If the OS provides a cluster file system that allows read/write access of a shared-disk partition between multiple nodes, the shared disk is mounted as such a cluster file system. Similarly, the system may use database software such as Oracle Parallel Server that permits multiple nodes running in a cluster to have concurrent read/write access to a shared disk. In such cases, a shared disk is already designed into the base OS and application software.

For operating systems where such shared access is not possible, because the OS and associated applications cannot manage a disk device shared with other nodes, the shared disk can be mounted as a read-only device. For many Web applications, having read-only access to Web related files is sufficient. For example, in Unix systems, a particular file system may be mounted as read-only.

#### MULTI-SWITCH COMPUTING GRID

The configuration described above in connection with FIG. 5 can be expanded to a large number of computing and storage nodes by interconnecting a plurality of VLAN switches to form a large switched VLAN fabric, and by interconnecting multiple SAN switches to form a large switched SAN mesh. In this case, a computing grid has the architecture generally shown in FIG. 5, except that the SAN/VLAN switched mesh contains a very large number of ports for CPUs and storage devices. A number of computing elements running the control plane can be physically connected to the control

WO 02/01202

PCT/US99/09653

ports of the VLAN/SAN switches, as described further below. Interconnection of multiple VLAN switches to create complex multi-campus data networks is known in this field. See, for example, G. Haviland, "Designing High-Performance Campus Networks with Multilayer Switching," Cisco Systems, Inc., and information available from Brocade.

#### SAN ARCHITECTURE

The description assumes that the SAN comprises Fibre-Channel switches and disk devices, and potentially Fibre-Channel edge devices such as SCSI-to-Fibre Channel bridges. However, SANs may be constructed using alternative technologies, such as Gigabit Ethernet switches, or switches that use other physical layer protocols. In particular, there are efforts currently underway to construct SANs over IP networks by running the SCSI protocol over IP. The methods and architecture described above is adaptable to these alternative methods of constructing a SAN. When a SAN is constructed by running a protocol like SCSI over IP over a VLAN capable layer 2 environment, then SAN zones are created by mapping them to different VLANs.

Also, Network Attached Storage (NAS) may be used, which works over LAN technologies such as fast Ethernet or Gigabit Ethernet. With this option, different VLANs are used in place of the SAN zones in order to enforce security and the logical partitioning of the computing grid. Such NAS devices typically support network file systems such as Sun's NFS protocol, or Microsoft's SMB, to allow multiple nodes to share the same storage.

#### CONTROL PLANE IMPLEMENTATION

As described herein, control planes may be implemented as one or more processing resources that are coupled to control and data ports of the SAN and VLAN switches. A variety of control plane implementations may be used and the invention is not limited to any particular control plane implementation. Various aspects of control plane implementation are described in more detail in the following sections: 1) control plane architecture; 2) master segment manager election; 3) administrative functions; and 4) policy and security considerations.

##### 1. Control Plane Architecture

WO 01/03263

PCT/00/01263

According to one embodiment, a control plane is implemented as a tree-like process hierarchy. The control process hierarchy generally includes one or more master segment manager mechanisms that are communicatively coupled to and control one or more slave segment manager mechanisms. The one or more slave segment manager mechanisms control one or more firm managers. The one or more firm managers manage one or more VSPs. The master and slave segment manager mechanisms may be implemented in hardware circuitry, computer software, or any combination thereof.

FIG. 9 is a block diagram 900 that illustrates a logical relationship between a control plane 902 and a computing grid 904 according to one embodiment. Control plane 902 controls and manages computing, networking and storage elements contained in computing grid 904 through special control plane or interface of the networking and storage elements in computing grid 904. Computing grid 904 includes a number of VSPs 906 or logical resource groups created in accordance with an embodiment as previously described herein.

According to one embodiment, control plane 902 includes a master segment manager 908, one or more slave segment managers 910 and one or more firm managers 912. Master segment manager 908, slave segment managers 910 and firm managers 912 may be co-located on a particular computing platform or may be distributed on multiple computing platforms. For purposes of explanation, only a single master segment manager 908 is illustrated and described, however, any number of master segment managers 908 may be employed.

Master segment manager 908 is communicatively coupled to, controls and manages slave segment managers 910. Each slave segment manager 910 is communicatively coupled to and manages one or more firm managers 912. According to one embodiment, each firm manager 912 is co-located on the same computing platform as the corresponding slave segment managers 910 with which it is communicatively coupled. Firm managers 912 establish, configure and maintain VSPs 906 on computing grid 904. According to one embodiment, each firm manager 912 is assigned a single VSP 906 to manage, however, firm managers 912 may also be assigned multiple VSPs 906. Firm managers 912 do not communicate directly with each other, but only through their respective slave segment managers 910. Slave segment managers 910 are responsible for monitoring the status of their assigned firm managers 912. Slave segment managers 910 react to any of their assigned firm managers 912 that have stalled or failed.

WO 02/03283

PCT/JP98/12953

Master segment manager 908 monitors the loading of VSFs 906 and determines an amount of resources to be allocated to each VSF 906. Master segment manager 908 then instructs slave segment managers 910 to allocate and de-allocate resources for VSFs 906 as appropriate through firm managers 912. A variety of load balancing algorithms may be implemented depending upon the requirements of a particular application and the invention is not limited to any particular load balancing approach.

Master segment manager 908 monitors loading information for the computing platforms on which slave segment managers 910 and firm managers 912 are executing to determine whether computing grid 904 is being adequately serviced. Master segment manager 908 allocates and de-allocates slave segment managers 910 and instructs slave segment managers 910 to allocate and de-allocate firm managers 912 as necessary to provide adequate management of computing grid 904. According to one embodiment, master segment manager 908 also manages the assignment of VSFs to firm managers 912 and the assignment of firm managers 912 to slave segment managers 910 as necessary to balance the load among firm managers 912 and slave segment managers 910. According to one embodiment, slave segment managers 910 actively communicate with master segment manager 908 and request changes to computing grid 904 and to request additional slave segment managers 910 and/or firm managers 912. If a processing platform fails on which one or more slave segment managers 910 and one or more firm managers 912 are executing, then master segment manager 908 reassigns the VSFs 906 from the firm managers 912 on the failed computing platform to other firm managers 912. In this situation, master segment manager 908 may also instruct slave segment managers 910 to initiate additional firm managers 912 to handle the reassignment of VSFs 906. Actively managing the number of computational resources allocated to VSFs 906, the number of active firm managers 912 and slave segment managers 910 allows overall power consumption to be controlled. For example, to conserve power master segment manager 908 may shutdown computing platforms that have no active slave segment managers 910 or firm managers 912. The power savings can be significant with large computing grids 904 and control planes 902.

According to one embodiment, master segment manager 908 manages slave segment managers 910 using a registry. The registry contains information about current slave segment managers 910 such as their state and assigned firm managers 912 and assigned VSFs 906. As slave segment managers 910 are allocated and de-allocated, the registry is updated to reflect the change in slave segment managers 910. For example,

WO 02/03262

PCT/US98/029653

when a new slave segment manager 910 is instantiated by master segment manager 908 and assigned one or more VSRs 906, the registry is updated to reflect the creation of the new slave segment manager 910 and its assigned firm managers 912 and VSRs 906. Master segment manager 908 may then periodically examine the registry to determine how to best assign VSRs 906 to slave segment managers 910.

According to one embodiment, the registry contains information about master segment manager 908 that can be accessed by slave segment managers 910. For example, the registry may contain data that identifies one or more active master segment managers 908 so that when a new slave segment manager 910 is created, the new slave segment manager 910 may check the registry to learn the identity of the one or more master segment managers 908.

The registry may be implemented in many forms and the invention is not limited to any particular implementation. For example, the registry may be a data file stored on a database 914 within control plane 902. The registry may instead be stored outside of control plane 902. For example, the registry may be stored on a storage device in computing grid 904. In this example, the storage device would be dedicated to control plane 902 and not allocated to VSRs 906.

## 2. Master Segment Manager Election

In general, a master segment manager is elected when a control plane is established or after a failure of an existing master segment manager. Although there is generally a single master segment manager for a particular control plane, there may be situations where it is advantageous to elect two or more master segment managers to co-manage the slave segment managers in the control plane.

According to one embodiment, slave segment managers in a control plane elect a master segment manager for that control plane. In the simple case where there is no master segment manager and only a single slave segment manager, then the slave segment manager becomes the master segment manager and allocates additional slave segment managers as needed. If there are two or more slave segment managers, then the two or more slave processes elect a new master segment manager by vote, e.g., by a quorum.

Since slave segment managers in a control plane are not necessarily persistent, particular slave segment managers may be selected to participate in a vote. For example, according to one embodiment, the register includes a timestamp for each slave segment

WO 02/03283

PCT/00/019653

manager that is periodically updated by each slave segment manager. The slave segment managers with timestamps that have been most recently updated, as determined according to specified selection criteria, are most likely to still be executing and are selected to vote for a new master segment manager. For example, a specified number of the most recent slave segment managers may be selected for a vote.

According to another embodiment, an election sequence number is assigned to all active slave segment managers and a new master segment manager is determined based upon the election sequence numbers for the active slave segment managers. For example, the lowest or highest election sequence number may be used to select a particular slave segment manager to be the next (or first) master segment manager.

Once a master segment manager has been established, the slave segment managers in the same control plane as the master segment manager periodically perform a health check on the master segment manager by contacting (ping) the current master segment manager to determine whether the master segment manager is still active. If a determination is made that the current master segment manager is no longer active, then a new master segment manager is elected.

FIG. 10 depicts a state diagram 1000 of a master segment manager election according to an embodiment. In state 1002, which is the slave segment manager main loop, the slave segment manager waits for the expiration of a ping timer. Upon expiration of the ping timer, state 1004 is entered. In state 1004, the slave segment manager pings the master segment manager. Also in state 1004, timeout (T0) for the slave segment manager is updated. If the master segment manager responds to the ping, then the master segment manager is still active and control returns to state 1002. If no response is received from the master segment manager after a specified period of time, then state 1006 is entered.

In state 1006, an active slave segment manager list is obtained and control proceeds to state 1008. In state 1008, a check is made to determine whether other slave segment managers have also not received a response from the master segment manager. Instead of sending messages to slave segment managers to make this determination, this information may be obtained from a database. If the slave segment managers do not agree that master segment manager is no longer active, i.e., one or more of the slave segment managers received a timely response from the master segment manager, then it is presumed that the current master segment manager is still active and control returns to state 1002. If a specified number of the slave segment managers have not received a

WFO 82/8263

PCT/IB00/29653

timely response from the current master segment manager, then it is assumed that the current master segment manager is "dead", i.e., no longer active, and control proceeds to state 1010.

In state 1010, the slave segment manager that initiated the process retrieves a current election number from an election table and the next election number from a database. The slave segment manager then updates the election table to include an entry that specifies the next election number and a unique address into a master election table. Control then proceeds to state 1012 where the slave segment manager reads the lowest sequence number for the current election number. In state 1014, a determination is made whether the particular slave segment manager has the lowest sequence number. If not, then control returns to state 1002. If so, then control proceeds to state 1016 where the particular slave segment manager becomes the master segment manager. Control then proceeds to state 1018 where the election number is incremented.

As described above, slave segment managers are generally responsible for servicing their assigned VSPs and allocating new VSPs in response to instructions from the master segment manager. Slave segment managers are also responsible for checking on the master segment manager and electing a new master segment manager if necessary.

FIG. 11 is a state diagram 1100 that illustrates various states of a slave segment manager according to an embodiment. Processing starts in a slave segment manager at state 1102. From state 1102, control proceeds to state 1104 in response to a request to confirm the state of the current master segment manager. In state 1104, the slave segment manager sends a ping to the current master segment manager to determine whether the current master segment manager is still active. If a timely response is received from the current master segment manager, the control proceeds to state 1106. In state 1106, a message is broadcasted to other slave segment managers to indicate that the master segment manager responded to the ping. From state 1106, control returns to state 1102.

In state 1104 if no timely master response is received, then control proceeds to state 1108. In state 1108, a message is broadcasted to other slave segment managers to indicate that the master segment manager did not respond to the ping. Control then returns to state 1102. Note that if a sufficient number of slave segment managers do not receive a response from the current master segment manager, then a new master segment manager is elected as described herein.

WO 03/01263

PC120000/2003

From start state 1102, control proceeds to state 1110 upon receipt of a request from the master segment manager to restart a VSF. In state 1110, a VSF is restarted and control returns to start state 1102.

As described above, a master segment manager is generally responsible for ensuring that VSFs in the computing grid controlled by the master segment manager are adequately serviced by one or more slave segment managers. To accomplish this, the master segment manager performs regular health checks on all slave segment managers in the same control plane as the master segment manager. According to one embodiment, master segment manager 908 periodically requests status information from slave segment managers 910. The information may include, for example, which VSFs 906 are being serviced by slave segment managers 910. If a particular slave segment manager 910 does not respond in a specified period of time, master segment manager 908 attempts to restart the particular slave segment manager 910. If the particular slave segment manager 910 cannot be restarted, then master segment manager 908 re-assigns the firm managers 912 from the failed slave segment manager 910 to another slave segment manager 910. Master segment manager 908 may then instantiate one or more additional slave segment managers 910 to re-balance the process loading. According to one embodiment, master segment manager 908 monitors the health of the computing platforms on which slave segment managers 910 are executing. If a computing platform fails, then master segment manager 908 reassigns the VSFs assigned to firm managers 912 on the failed computing platform to firm managers 912 on another computing platform.

FIG. 12 is a state diagram 1200 for a master segment manager. Processing starts in a master segment manager start state 1202. From state 1202, control proceeds to state 1204 when master segment manager 908 makes a periodic health check or request to slave segment managers 910 in control plane 902. From state 1204, if all slave segment managers 910 respond as expected, then control returns to state 1202. This occurs if all slave segment managers 910 provide the specified information to master segment manager 908, indicating that all slave segment managers 910 are operating normally. If one or more slave segment managers 910 either don't respond, or the response otherwise indicates that one or more slave segment managers 910 have failed, then control proceeds to state 1206.

In state 1206, master segment manager 908 attempts to restart the failed slave segment managers 910. This may be accomplished in several ways. For example, master segment manager 908 may send a restart message to a non-responsive or failed slave

WO 01/43103

PCYDRA/0963

segment manager 910. From state 1206, if all slave segment managers 910 respond as expected, i.e., have been successfully restarted, then control returns to state 1202. For example, when a failed slave segment manager 910 is successfully restarted, the slave segment manager 910 sends a restart confirmation message to master segment manager 908. From state 1206, if one or more slave segment managers have not been successfully restarted, then control proceeds to state 1208. This situation may occur if master segment manager 908 does not receive a restart confirmation message from a particular slave segment manager 910.

In state 1208, master segment manager 908 determines the current loading of the machines on which slave segment managers 910 are executing. To obtain the slave segment manager 908 loading information, master segment manager 908 polls slave segment managers 910 directly or obtains the loading information from another location, for example from database 914. The invention is not limited to any particular approach for master segment manager 908 to obtain the loading information for slave segment managers 910.

Control then proceeds to state 1210 where the VSFs 906 assigned to the failed slave segment managers 910 are re-assigned to other slave segment managers 910. The slave segment managers 910 to which the VSFs 906 are assigned inform master segment manager 908 when the reassignment has been completed. For example, slave segment managers 910 may send a reassignment confirmation message to master segment manager 908 to indicate that the reassignment of VSFs 906 has been successfully completed. Control remains in state 1210 until reassignment of all VSFs 906 associated with the failed slave segment managers 910 has been confirmed. Once confirmed, control returns to state 1202.

Instead of reassigning VSFs 906 associated with a failed slave segment manager 910 to other active slave segment managers 910, master segment manager 908 may allocate additional slave segment managers 910 and then assign those VSFs 906 to the new slave segment managers 910. The choice of whether to reassign VSFs 906 to existing slave segment managers 910 or to new slave segment managers 910 depends, at least in part, on latency associated with allocating new slave segment managers 910 and latency associated with reassigning VSFs 906 to an existing slave segment manager 910. Either approach may be used depending upon the requirements of a particular application and the invention is not limited to either approach.

WO 02/03063

FCT/USM/29853

### 3. Administrative Functions

According to one embodiment, control plane 902 is communicatively coupled to a global grid manager. Control plane 902 provides billing, fault, capacity, loading and other computing grid information to the global grid manager. FIG. 13 is a block diagram 1300 that illustrates the use of a global grid manager according to an embodiment.

In FIG. 13, a computing grid 1300 is partitioned into logical portions called grid segments 1302. Each grid segment 1302 includes a control plane 902 that controls and manages a data plane 904. In this example, each data plane 904 is the same as the computing grid 904 of FIG. 9, but are referred to as "data planes" to illustrate the use of a global grid manager to manage multiple control planes 902 and data planes 904, i.e., grid segments 1302.

Each grid segment is communicatively coupled to a global grid manager 1304. Global grid manager 1304, control planes 902 and computing grids 904 may be co-located on a single computing platform, or may be distributed across multiple computing platforms and the invention is not limited to any particular implementation.

Global grid manager 1304 provides centralized management and services for any number of grid segments 1302. Global grid manager 1304 may collect billing, loading and other information from control planes 902 used in a variety of administrative tasks. For example, the billing information is used to bill for services provided by computing grids 904.

### 4. Policy and Security Considerations

As described herein, a slave segment manager in a control plane must be able to communicate with its assigned VSPs in a computing grid. Similarly, VSPs in a computing grid must be able to communicate with their assigned slave segment manager. Further, VSPs in a computing grid must not be allowed to communicate with each other to prevent one VSP from in any way causing a change in the configuration of another VSP. Various approaches for implementing these policies are described hereinafter.

FIG. 14 is a block diagram 1400 of an architecture for connecting a control plane to a computing grid according to an embodiment. Control ("CTL") ports of VLAN switches (VLAN SW1 through VLAN SWn), collectively identified by reference numeral 1402, and SAN switches (SAN SW1 through SAN SWn), collectively identified by reference numeral 1404, are connected to an Ethernet subnet 1406. Ethernet subnet 1406 is connected to a plurality of computing elements (CPU1, CPU2 through CPUm), that are

WG 0283203

PCT/US01/79053

collectively identified by reference numeral 1408. Thus, only computing elements of control plane 1408 are communicatively coupled to the control ports (CTL) of VLAN switches 1402 and SAN switches 1404. This configuration prevents computing elements in a VSP (not illustrated), from changing the membership of the VLANs and SAN zones associated with itself or any other VSP. This approach is also applicable to situations where the control ports are serial or parallel ports. In these situations, the ports are coupled to the control plane 1408 computing elements.

FIG. 15 is a block diagram 1500 of a configuration for connecting control plane computing elements (CP CPU1, CP CPU2 through CP CPU<sub>n</sub>) 1502 to data ports according to an embodiment. In this configuration, control plane computing elements 502 periodically send a packet to a control plane agent 1504 that acts on behalf of control plane computing elements 1502. Control plane agent 1504 periodically polls computing elements 502 for real-time data and sends the data to control plane computing elements 1502. Each segment manager in control plane 1502 is communicatively coupled to a control plane (CP) LAN 1506. CP LAN 1506 is communicatively coupled to a special port V17 of VLAN Switch 504 through a CP firewall 1508. This configuration provides a scalable and secure means for control plane computing elements 1502 to collect real-time information from computing elements 502.

FIG. 16 is a block diagram 1600 of an architecture for connecting a control plane to a computing grid according to an embodiment. A control plane 1602 includes control plane computing elements CP CPU1, CP CPU2 through CP CPU<sub>n</sub>. Each control plane computing element CP CPU1, CP CPU2 through CP CPU<sub>n</sub> in control plane 1602 is communicatively coupled to a port S1, S2 through S<sub>n</sub> of a plurality of SAN switches that collectively form a SAN mesh 1604.

SAN mesh 1604 includes SAN ports So, Sp that are communicatively coupled to storage devices 1606 that contain data that is private to control plane 1602. Storage devices 1606 are depicted in FIG. 16 as disks for purposes of explanation. Storage devices 1606 may be implemented by any type of storage medium, and the invention is not limited to any particular type of storage medium for storage devices 1606. Storage devices 1606 are logically located in a control plane private storage zone 1608. Control plane private storage zone 1608 is an area where control plane 1602 maintains log files, statistical data, current control plane configuration information and software that implements control plane 1602. SAN ports So, Sp are only part of the control plane private storage zone and are never placed on any other SAN zone so that only computing

WD 62/63/63

PCT/JP00/01953

elements in control plane 1602 can access the storage devices 1606. Furthermore, ports S1, S2 through S6, S0 and S7 are in a control plane SAN zone that may only be communicatively coupled to computing elements in control plane 1602. These ports are not accessible by computing elements in VSPs (not illustrated).

According to one embodiment, when a particular computing element CP CPU1, CP CPU2 through CP CPU6 needs to access a storage device, or a portion thereof, that is part of a particular VSP, the particular computing element is placed into the SAN zone for the particular VSP. For example, suppose that computing element CP CPU 2 needs to access VSP1 disks 1616. In this situation, port S2, which is associated with control plane CP CPU 2, is placed in the SAN zone of VSP1, which includes port S6. Once computing element CP CPU2 is done accessing the VSP1 disks 1616 on port S6, computing element CP CPU2 is removed from the SAN zone of VSP1.

Similarly, suppose computing element CP CPU 1 needs to access VSP2 disks 1612. In this situation, computing element CP CPU1 is placed in the SAN zone associated with VSP2. As a result, port S1 is placed in the SAN zone associated with VSP2, which includes the zone containing port S3. Once computing element CP CPU1 is done accessing the VSP2 disks 1612 connected to port S3, computing element CP CPU1 is removed from the SAN zone associated with VSP2. This approach ensures the integrity of control plane computing elements and the control plane storage zones 1608 by tightly controlling access to resources using tight SAN zone control.

As previously described, a single control plane computing element may be responsible for managing several VSPs. Accordingly, a single control plane computing element must be capable of manifesting itself in multiple VSPs simultaneously, while enforcing firewalling between the VSPs according to policy rules established for each control plane. Policy rules may be stored in database 914 (FIG. 9) of each control plane or implemented by control segment manager 1302 (FIG. 13).

According to one embodiment, tight binding between VLAN tagging and IP addresses are used to prevent spoofing attacks by a VSP since (physical switch) port-based VLAN tags are not spoofable. An incoming IP packet on a given VLAN interface must have the same VLAN tag and IP address as the logical interface on which the packet arrives. This prevents IP spoofing attacks where a malicious server in a VSP spoofs the source IP address of a server in another VSP and potentially modifies the logical structure of another VSP or otherwise subverts the security of computing grid functions.

WO 01/43203

PC/H086/19653

Circumventing this VLAN tagging approach requires physical access to the computing grid which can be prevented using high security (Class A) data centers.

A variety of network frame tagging formats may be used to tag data packets and the invention is not limited to any particular tagging format. According to one embodiment, IEEE 802.1q VLAN tags are used, although other formats may also be suitable. In this example, a VLANID address consistency check is performed at a subsystem in the IP stack where 802.1q tag information is present to control access. In this example, computing elements are configured with a VLAN capable network interface card (NIC) in a manner that allows the computing elements to be communicatively coupled to multiple VLANs simultaneously.

FIG. 17 is a block diagram 1700 of an arrangement for enforcing tight binding between VLAN tags and IP addresses according to an embodiment. Computing elements 1702 and 1704 are communicatively coupled to ports v1 and v2 of a VLAN switch 1706 via NICs 1708 and 1710, respectively. VLAN switch 1706 is also communicatively coupled to access switches 1712 and 1714. Ports v1 and v2 are configured in tagged mode. According to one embodiment, IEEE 802.1q VLAN tag information is provided by VLAN switch 1706.

#### A WIDE AREA COMPUTING GRID

The VSF described above can be distributed over a WAN in several ways.

In one alternative, a wide area backbone may be based on Asynchronous Transfer Mode (ATM) switching. In this case, each local area VLAN is extended into a wide area using Emulated LANs (ELANs) which are part of the ATM LAN Emulation (LANE) standard. In this way, a single VSF can span across several wide area links, such as ATM/SONET/OC-12 links. An ELAN becomes part of a VLAN which extends across the ATM WAN.

Alternatively, a VSF is extended across a WAN using a VPN system. In this embodiment, the underlying characteristics of the network become irrelevant, and the VPN is used to interconnect two or more VSFs across the WAN to make a single distributed VSF.

Data mirroring technologies can be used in order to have local copies of the data in a distributed VSF. Alternatively, the SAN is bridged over the WAN using one of several SAN to WAN bridging techniques, such as SAN-to-ATM bridging or SAN-to-

WO 02/03282

PCIDENR/1965

Gigabit Ethernet bridging. SANs constructed over IP networks naturally extend over the WAN since IP works well over such networks.

FIG. 18 is a block diagram of a plurality of VSFs extended over WAN connections. A San Jose Center, New York Center, and London center are coupled by WAN connections. Each WAN connection comprises an ATM, LAN, or VPN connection in the manner described above. Each center comprises at least one VSF and at least one Mta Pool. For example, the San Jose center has VSF1A and Mta Pool A. In this configuration, the computing resources of each Mta Pool of a center are available for allocation or assignment to a VSF located in any other center. When such allocation or assignment is carried out, a VSF becomes extended over the WAN.

#### EXAMPLE USES OF VSFs

The VSF architecture described in the examples above may be used in the context of Web server systems. Thus, the foregoing examples have been described in terms of Web servers, application servers and database servers constructed out of the CPUs in a particular VSF. However, the VSF architecture may be used in many other computing contexts and to provide other kinds of servicing; it is not limited to Web server systems.

#### — A DISTRIBUTED VSF AS PART OF A CONTENT DISTRIBUTION NETWORK

In one embodiment, a VSF provides a Content Distribution Network (CDN) using a wide area VSF. The CDN is a network of caching servers that performs distributed caching of data. The network of caching servers may be implemented, for example, using TrafficServer (TS) software commercially available from Internet Corporation, San Mateo, California. TS is a cluster aware system; the system scales as more CPUs are added to a set of caching Traffic Server computing elements. Accordingly, it is well suited to a system in which adding CPUs is the mechanism for scaling upwards.

In this configuration, a system can dynamically add more CPUs to that portion of a VSF that runs caching software such as TS, thereby growing the cache capacity at a point close to where heavy Web traffic is occurring. As a result, a CDN may be constructed that dynamically scales in CPU and I/O bandwidth in an adaptive way.

WO 02/03203

PCEN0001/19653

#### — A VSF FOR HOSTED INTRANET APPLICATIONS

There is growing interest in offering intranet applications such as Enterprise Resource Planning (ERP), CRM and CRM software as hosted and managed services. Technologies such as Citrix WinFrame and Citrix MetaFrame allow an enterprise to provide Microsoft Windows applications as a service on a thin client such as a Windows CE device or Web browser. A VSF can host such applications in a scalable manner.

For example, the SAP R/3 ERP software, commercially available from SAP Aktiengesellschaft of Germany, allows an enterprise to load balance using multiple Application and Database Servers. In the case of a VSF, an enterprise would dynamically add more Application Servers (e.g., SAP Dialog Servers) to a VSF in order to scale up the VSF based on real-time demand or other factors.

Similarly, Citrix MetaFrame allows an enterprise to scale up Windows application usage on a server farm running the hosted Windows applications by adding more Citrix servers. In this case, for a VSF, the Citrix MetaFrame VSF would dynamically add more Citrix servers in order to accommodate more users of MetaFrame hosted Windows applications. It will be apparent that many other applications may be hosted in a manner similar to the illustrative examples described above.

#### — CUSTOMER INTERACTION WITH A VSF

Since a VSF is created on demand, a VSF customer or organization that "owns" the VSF may interact with the system in various ways in order to customize a VSF. For example, because a VSF is created and modified instantly via the control plane, the VSF customer may be granted privileged access to create and modify its VSF itself. The privileged access may be provided using password authentication provided by Web pages and security applications, token card authentication, Kerberos exchange, or other appropriate security elements.

In one exemplary embodiment, a set of Web pages are served by the computing element, or by a separate server. The Web pages enable a customer to create a certain VSF, by specifying a number of tiers, the number of computing elements in a particular tier, the hardware and software platform used for each element, and things such as what kind of Web server, application server, or database server software should be pre-configured on those computing elements. Thus, the customer is provided with a virtual provisioning console.

WO 00/23263

PCT/JP99/00653

After the customer or user enters such provisioning information, the control plane parses and evaluates the order and queues it for execution. Orders may be reviewed by human managers to ensure that they are appropriate. Credit checks of the enterprise may be run to ensure that it has appropriate credit to pay for the requested services. If the provisioning order is approved, the control plane may configure a VSF that matches the order, and return to the customer a password providing root access to one or more of the computing elements in the VSF. The customer may then upload master copies of applications to execute in the VSF.

When the enterprise that hosts the computing grid is a for-profit enterprise, the Web page may also receive payment related information, such as a credit card, a PO number, electronic check, or other payment method.

In another embodiment, the Web pages enable the customer to choose one of several VSF service plans, such as automatic growth and shrinkage of a VSF between a minimum and maximum number of elements, based on real-time load. The customer may have a control value that allows the customer to change parameters such as minimum number of computing elements in a particular tier such as Web servers, or a time period in which the VSF must have a minimal amount of server capacity. The parameters may be linked to billing software that would automatically adjust the customer's bill rate and generate billing log file entries.

Through the privileged access mechanism the customer can obtain reports and monitor real-time information related to usage, load, bits or transactions per second, and adjust the characteristics of a VSF based on the real-time information. It will be apparent that the foregoing features offer significant advantages over conventional manual approaches to constructing a server farm. In the conventional approaches, a user cannot automatically influence server farm's properties without going through a cumbersome manual procedure of adding servers and configuring the server farm in various ways.

#### - BILLING MODELS FOR A VSF

Given the dynamic nature of a VSF, the enterprise that hosts the computing grid and VSFs may bill service fees to customers who own VSFs using a billing model for a VSF which is based on actual usage of the computing elements and storage elements of a VSF. It is not necessary to use a flat fee billing model. The VSF architecture and methods disclosed herein enable a "pay-as-you-go" billing model because the resources of a given VSF are not statically assigned. Accordingly, a particular customer having a

WO 01/01283

PCT/US99/02653

highly variable usage load on its server firm could save money because it would not be billed a rate associated with constant peak server capacity, but rather, a rate that reflects a running average of usage, instantaneous usage, etc.

For example, an enterprise may operate using a billing model that stipulates a flat fee for a minimum number of computing elements, such as 10 servers, and stipulates that when real-time load requires more than 10 elements, then the user is billed at an incremental rate for the extra servers, based on how many extra servers were needed and for the length of time that they are needed. The units of such bills may reflect the resources that are billed. For example, bills may be expressed in units such as MIPS-hours, CPU-hours, thousands of CPU seconds, etc.

#### - A CUSTOMER VISIBLE CONTROL PLANE API

In another alternative, the capacity of a VSF may be controlled by providing the customer with an application programming interface (API) that defines calls to the control plane for changing resources. Thus, an application program prepared by the customer could issue calls or requests using the API to ask for more servers, more storage, more bandwidth, etc. This alternative may be used when the customer needs the application program to be aware of the computing grid environment and to take advantage of the capabilities offered by the control plane.

Nothing in the above-disclosed architecture requires the customer to modify its application for use with the computing grid. Existing applications continue to work as they do in manually configured server firms. However, an application can take advantage of the dynamism possible in the computing grid, if it has a better understanding of the computing resources it needs based on the real-time load monitoring functions provided by the control plane. An API of the foregoing nature, which enables an application program to change the computing capacity of a server firm, is not possible using existing manual approaches to constructing a server firm.

#### - AUTOMATIC UPDATING AND VERSIONING

Using the methods and mechanisms disclosed herein, the control plane may carry out automatic updating and versioning of operating system software that is executed in computing elements of a VSF. Thus, the end user or customer is not required to worry about updating the operating system with a new patch, bug fix, etc. The control plane can

W/O 02A/203

PCT/JP01/01653

maintain a library of such software elements as they are received and automatically distribute and install them in computing elements of all affected V2Ns.

#### IMPLEMENTATION MECHANISMS

The computing elements and control phase may be implemented in several forms and the invention is not limited to any particular form. In one embodiment, each computing element is a general purpose digital computer having the elements shown in FIG. 19 except for non-volatile storage device 1910, and the control phase is a general purpose digital computer of the type shown in FIG. 19 operating under control of program instructions that implement the processes described herein.

Figure 19 is a block diagram that illustrates a computer system 1900 upon which an embodiment of the invention may be implemented. Computer system 1900 includes a bus 1902 or other communication mechanism for communicating information, and a processor 1904 coupled with bus 1902 for processing information. Computer system 1900 also includes a main memory 1906, such as a random access memory (RAM) or other dynamic storage device, coupled to bus 1902 for storing information and instructions to be executed by processor 1904. Main memory 1906 also may be used for storing temporary variables or other intermediate information during execution of instructions to be executed by processor 1904. Computer system 1900 further includes a read only memory (ROM) 1908 or other static storage device coupled to bus 1902 for storing static information and instructions for processor 1904. A storage device 1910, such as a magnetic disk or optical disk, is provided and coupled to bus 1902 for storing information and instructions.

Computer system 1900 may be coupled via bus 1902 to a display 1912, such as a cathode ray tube (CRT), for displaying information to a computer user. An input device 1914, including alphanumeric and other keys, is coupled to bus 1902 for communicating information and command selections to processor 1904. Another type of user input device is cursor control 1916, such as a mouse, a trackball, or cursor direction keys for communicating direction information and command selections to processor 1904 and for controlling cursor movement on display 1912. This input device typically has two degrees of freedom in two axes, a first axis (e.g., x) and a second axis (e.g., y), that allows the device to specify positions in a plane.

The invention is related to the use of computer system 1900 for controlling an extensible computing system. According to one embodiment of the invention, controlling

WO 01/02282

PCT/JP99/00653

an extensible computing system is provided by computer system 1900 in response to processor 1904 executing one or more sequences of one or more instructions contained in main memory 1906. Such instructions may be read into main memory 1906 from another computer-readable medium, such as storage device 1910. Recreational of the sequences of instructions contained in main memory 1906 causes processor 1904 to perform the process steps described herein. One or more processors in a multi-processing arrangement may also be employed to execute the sequences of instructions contained in main memory 1906. In alternative embodiments, hard-wired circuitry may be used in place of or in combination with software instructions to implement the invention. Thus, embodiments of the invention are not limited to any specific combination of hardware circuitry and software.

The term "computer-readable medium" as used herein refers to any medium that participates in providing instructions to processor 1904 for execution. Such a medium may take many forms, including but not limited to, non-volatile media, volatile media, and transmission media. Non-volatile media includes, for example, optical or magnetic disks, such as storage device 1910. Volatile media includes dynamic memory, such as main memory 1906. Transmission media includes coaxial cables, copper wire and fiber optics, including the wires that comprise bus 1902. Transmission media can also take the form of acoustic or light waves, such as those generated during radio wave and infrared data communications.

Common forms of computer-readable media include, for example, a floppy disk, a flexible disk, hard disk, magnetic tape, or any other magnetic medium, a CD-ROM, any other optical medium, punch cards, paper tape, any other physical medium with patterns of holes, a RAM, a PROM, and EPROM, a FLASH-EPROM, any other memory chip or cartridge, a carrier wave as described hereinafter, or any other medium from which a computer can read.

Various forms of computer readable media may be involved in carrying one or more sequences of one or more instructions to processor 1904 for execution. For example, the instructions may initially be carried on a magnetic disk of a remote computer. The remote computer can load the instructions into its dynamic memory and send the instructions over a telephone line using a modem. A modem local to computer system 1900 can receive the data on the telephone line and use an infrared transmitter to convert the data to an infrared signal. An infrared detector coupled to bus 1902 can receive the data carried in the infrared signal and place the data on bus 1902. Bus 1902

WO 2004/02923

PC20030419653

carries the data to main memory 1906, from which processor 1904 retrieves and executes the instructions. The instructions received by main memory 1906 may optionally be stored on storage device 1910 either before or after execution by processor 1904.

Computer system 1900 also includes a communication interface 1918 coupled to bus 1902. Communication interface 1918 provides a two-way data communication coupling to a network link 1920 that is connected to a local network 1922. For example, communication interface 1918 may be an integrated services digital network (ISDN) card or a modem to provide a data communication connection to a corresponding type of telephone line. As another example, communication interface 1918 may be a local area network (LAN) card to provide a data communication connection to a compatible LAN. Wireless links may also be implemented. In any such implementation, communication interface 1918 sends and receives electrical, electromagnetic or optical signals that carry digital data streams representing various types of information.

Network link 1920 typically provides data communication through one or more networks to other data devices. For example, network link 1920 may provide a connection through local network 1922 to a host computer 1924 or to data equipment operated by an Internet Service Provider (ISP) 1926. ISP 1926 in turn provides data communication services through the worldwide packet data communication network now commonly referred to as the "Internet" 1928. Local network 1922 and Internet 1928 both use electrical, electromagnetic or optical signals that carry digital data streams. The signals through the various networks and the signals on network link 1920 and through communication interface 1918, which carry the digital data to and from computer system 1900, are exemplary forms of carrier waves transporting the information.

Computer system 1900 can send messages and receive data, including program code, through the network(s), network link 1920 and communication interface 1918. In the Internet example, a server 1930 might transmit a requested code for an application program through Internet 1928, ISP 1926, local network 1922 and communication interface 1918. In accordance with the invention, one such downloaded application provides for controlling an extensible computing system as detailed herein.

The received code may be executed by processor 1904 as it is received, and/or stored in storage device 1910, or other non-volatile storage for later execution. In this manner, computer system 1900 may obtain application code in the form of a carrier wave.

The computing grid disclosed herein may be compared conceptually to the public electric power network that is sometimes called the power grid. The power grid provides

WO 02/01281

PC92006/09953

a scalable means for many parties to obtain power services through a single wide-scale power infrastructure. Similarly, the computing grid disclosed herein provides computing services to many organizations using a single wide-scale computing infrastructure. Using the power grid, power consumers do not independently manage their own personal power equipment. For example, there is no reason for a utility consumer to run a personal power generator at its facility, or in a shared facility and manage its capacity and growth on an individual basis. Instead, the power grid enables the wide-scale distribution of power to vast segments of the population, thereby providing great economies of scale. Similarly, the computing grid disclosed herein can provide computing services to vast segments of the population using a single wide-scale computing infrastructure.

In the foregoing specification, the invention has been described with reference to specific embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention. The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense.

WO 02/02063

PCT/JP00/0063

## CLAIMS

What is claimed is:

1. A control apparatus comprising:  
a master control mechanism; and  
one or more slave control mechanisms communicatively coupled to the master control mechanism and being configured to, in response to one or more instructions from the master control mechanism, establish a first logical resource group that contains a first subset of processing resources and a first subset of storage resources by:  
selecting the first subset of processing resources from a set of processing resources,  
selecting the first subset of storage resources from a set of storage resources, and  
causing the first subset of processing resources to be communicatively coupled to the first subset of storage resources.
2. A control apparatus as recited in Claim 1, wherein the master control mechanism is a master control process executing on one or more processors and the one or more slave control mechanisms are one or more slave processes executing on the one or more processors.
3. A control apparatus as recited in Claim 1, wherein the master control mechanism is one or more master processors and the one or more slave control mechanisms are one or more slave processors.
4. A control apparatus as recited in Claim 1, wherein the master control mechanism is configured to, based upon slave control process mechanism loading, dynamically reassign control, between the one or more slave control mechanisms, of one or more processing resources from the subset of processing resources and one or more storage resources from the subset of storage resources.
5. A control apparatus as recited in Claim 1, wherein the master control mechanism is configured to, based upon slave control process mechanism loading,

WO 01/43203

PCT/US99/29653

- dynamically allocate one or more additional slave control mechanisms, and assign control of one or more processing resources from the subset of processing resources and one or more storage resources from the subset of storage resources to the one or more additional slave control mechanisms.
6. A control apparatus as recited in Claim 1, wherein the master control mechanism is configured to, based upon slave control process mechanism loading, reassign control to one or more other slave control mechanisms from the one or more slave control mechanisms of one or more particular processing resources from the subset of processing resources and one or more particular storage resources from the subset of storage resources that were previously assigned to one or more particular slave control mechanisms from the one or more slave control mechanisms, and dynamically de-allocate the one or more particular slave control mechanisms.
7. A control apparatus as recited in Claim 1, wherein the master control mechanism is configured to:
- determine a status of the one or more slave control mechanisms;
  - if one or more particular slave control mechanisms from the one or more slave control mechanisms are not responding or functioning correctly, then attempting to restart the one or more particular slave control mechanisms, and
  - if the one or more particular slave control mechanisms cannot be restarted, then
    - initiating one or more new slave control mechanisms, and
    - reassigning control of processing resources and storage resources from the one or more particular slave control mechanisms to the one or more new slave control mechanisms.
8. A control apparatus as recited in Claim 1, wherein the one or more slave control mechanisms are configured to:
- determine a status of the master control mechanism, and

W/O 4147263

PC7Y0086/12623

if the master control mechanism has failed or is no longer functioning properly, select a new master control mechanism from the one or more slave control mechanisms.

9. A control apparatus as recited in Claim 1, wherein the one or more instructions from the master control mechanism are generated based upon expected processing and storage requirements for the first logical resource group.
10. A control apparatus as recited in Claim 1, wherein the one or more slave control mechanisms are further configured to, in response to the one or more instructions from the master control mechanism, perform the following:  
dynamically change the number of processing resources in the first subset of processing resources,  
dynamically change the number of storage resources in the first subset of storage resources,  
dynamically change the communicative coupling between the first subset of processing resources and the first subset of storage resources to reflect changes in the number of processing resources in the first subset of processing resources and the number of storage resources in the first subset of storage resources.
11. A control apparatus as recited in Claim 10, wherein changes to the number of processing resources in the first subset of processing resources and the number of storage resources in the first subset of storage resources is instructed by the master control mechanism based upon actual loading of the first subset of processing resources and first subset of storage resources.
12. A control apparatus as recited in Claim 1, wherein the one or more slave control mechanisms are further configured to, in response to the one or more instructions from the master control mechanism, establish a second logical resource group that contains a second subset of processing resources and a second subset of storage resources, wherein the second logical resource group is communicatively isolated from the first logical resource group, by:

WO 01/61263

PCT/JP99/01953

selecting the second subset of processing resources from the set of processing resources,  
 selecting the second subset of storage resources from the set of processing resources, and  
 causing the second subset of processing resources to be communicatively coupled to the second subset of storage resources.

13. A control apparatus as recited in Claim 12, wherein:
  - the first subset of processing resources is communicatively coupled to the first subset of storage resources using one or more storage area network (SAN) switches,
  - the second subset of processing resources is communicatively coupled to the second subset of storage resources using the one or more SAN switches, and
  - the second logical resource group is communicatively isolated from the first logical resource group using tagging and SAN zoning.
14. A control apparatus as recited in Claim 13, wherein SAN zoning is performed using port-level SAN zoning or LUN level SAN zoning.
15. A control apparatus as recited in Claim 1, wherein:
  - the master control mechanism is communicatively coupled to a central control mechanism,
  - the master control mechanism is configured to provide loading information for the first logical resource group to the central control mechanism, and
  - the master control mechanism is further configured to generate the one or more instructions for the one or more slave control mechanisms based upon one or more control control instructions received from the central control mechanism.
16. A method for managing processing resources comprising the steps of:
  - initiating a master control mechanism; and
  - initiating one or more slave control mechanisms communicatively coupled to the master control mechanism and being configured to, in response to one or

WO 01/03202

PCT/US00/19853

move instructions from the master control mechanism, establish a first logical resource group that contains a first subset of processing resources and a first subset of storage resources by:

- selecting the first subset of processing resources from a set of processing resources,
- selecting the first subset of storage resources from a set of storage resources, and
- coupling the first subset of processing resources to be communicatively coupled to the first subset of storage resources.

17. A method as recited in Claim 16, wherein:
  - initiating a master control mechanism includes initiating a master control process executing on one or more processors, and
  - initiating one or more slave control mechanisms includes initiating one or more slave processes executing on the one or more processors.
18. A method as recited in Claim 16, wherein:
  - initiating a master control mechanism includes initiating one or more master control processors, and
  - initiating one or more slave control mechanisms includes initiating one or more slave processors.
19. A method as recited in Claim 16, further comprising the master control mechanism dynamically reassigning control, based upon slave control process mechanism loading, between the one or more slave control mechanisms, of one or more processing resources from the subset of processing resources and one or more storage resources from the subset of storage resources.
20. A method as recited in Claim 16, further comprising the master control mechanism, based upon slave control process mechanism loading, dynamically allocating one or more additional slave control mechanisms, and assigning control of one or more processing resources from the subset of processing resources and one or more storage resources from the subset of storage resources to the one or more additional slave control mechanisms.

WO 02/03203

PC/P2001/0963

21. A method as recited in Claim 16, further comprising the master control mechanism, based upon slave control process mechanism loading, reassigning control to one or more other slave control mechanisms from the one or more slave control mechanisms of one or more particular processing resources from the subset of processing resources and one or more particular storage resources from the subset of storage resources that were previously assigned to one or more particular slave control mechanisms from the one or more slave control mechanisms, and dynamically de-allocating the one or more particular slave control mechanisms.
22. A method as recited in Claim 16, further comprising the master control mechanism:  
determining a status of the one or more slave control mechanisms,  
if one or more particular slave control mechanisms from the one or more slave control mechanisms are not responding or functioning correctly, then attempting to restart the one or more particular slave control mechanisms, and  
if the one or more particular slave control mechanisms cannot be restarted, then  
isolating one or more new slave control mechanisms, and  
reassigning control of processing resources and storage resources from the one or more particular slave control mechanisms to the one or more new slave control mechanisms.
23. A method as recited in Claim 16, further comprising the one or more slave control mechanisms:  
determining a status of the master control mechanism, and  
if the master control mechanism has failed or is no longer functioning properly, selecting a new master control mechanism from the one or more slave control mechanisms.

WO 03/02823

PCT/JP00/2963

24. A method as recited in Claim 16, wherein the one or more instructions from the master control mechanism are generated based upon expected processing and storage requirements for the first logical resource group.
25. A method as recited in Claim 16, further comprising the one or more slave control mechanisms, in response to the one or more instructions from the master control mechanism, performing the following:  
dynamically changing the number of processing resources in the first subset of processing resources,  
dynamically changing the number of storage resources in the first subset of storage resources,  
dynamically changing the communicative coupling between the first subset of processing resources and the first subset of storage resources to reflect changes in the number of processing resources in the first subset of processing resources and the number of storage resources in the first subset of storage resources.
26. A method as recited in Claim 25, wherein changes to the number of processing resources in the first subset of processing resources and the number of storage resources in the first subset of storage resources is instructed by the master control mechanism based upon actual loading of the first subset of processing resources and first subset of storage resources.
27. A method as recited in Claim 16, further comprising the one or more slave control mechanisms, in response to the one or more instructions from the master control mechanism, establishing a second logical resource group that contains a second subset of processing resources and a second subset of storage resources, wherein the second logical resource group is communicatively isolated from the first logical resource group, by:  
selecting the second subset of processing resources from the set of processing resources,  
selecting the second subset of storage resources from the set of processing resources, and

WO 02/03263

PC/DEN/02/0063

coupling the second subset of processing resources to be communicatively coupled to the second subset of storage resources.

28. A method as recited in Claim 27, wherein:
  - the first subset of processing resources is communicatively coupled to the first subset of storage resources using one or more storage area network (SAN) switches,
  - the second subset of processing resources is communicatively coupled to the second subset of storage resources using the one or more SAN switches, and
  - the second logical resource group is communicatively isolated from the first logical resource group using tagging and SAN zoning.
29. A method as recited in Claim 28, wherein SAN zoning is performed using port-level SAN zoning or LUN level SAN zoning.
30. A method as recited in Claim 16, wherein:
  - the master control mechanism is communicatively coupled to a central control mechanism,
  - the master control mechanism is configured to provide loading information for the first logical resource group to the central control mechanism, and
  - the master control mechanism is further configured to generate the one or more instructions for the one or more slave control mechanisms based upon one or more control instructions received from the central control mechanism.
31. A computer-readable medium carrying one or more sequences of one or more instructions for managing processing resources, wherein execution of the one or more sequences of one or more instructions by one or more processors causes the one or more processors to perform the steps of:
  - initiating a master control mechanism; and
  - initiating one or more slave control mechanisms communicatively coupled to the master control mechanism and being configured to, in response to one or more instructions from the master control mechanism, establish a first

WO 00/41283

PCT/JP99/00633

logical resource group that contains a first subset of processing resources  
and a first subset of storage resources by:  
selecting the first subset of processing resources from a set of processing  
resources,  
selecting the first subset of storage resources from a set of storage  
resources, and  
causing the first subset of processing resources to be communicatively coupled to  
the first subset of storage resources.

WO 02/03280

PCT/JP02/00453

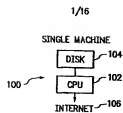


FIG. 1A

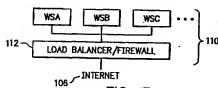


FIG. 1B

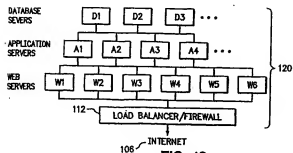


FIG. 1C

WO 02/0283

PC/EDM/2853

2/16

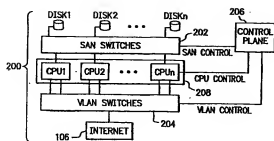


FIG. 2

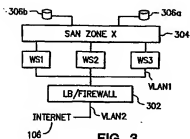
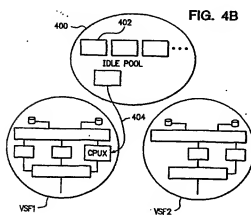
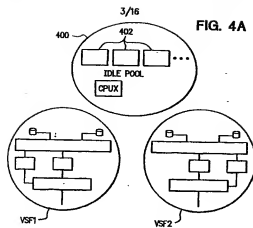


FIG. 3

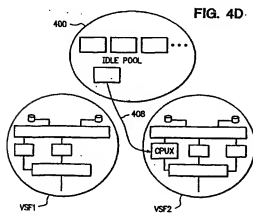
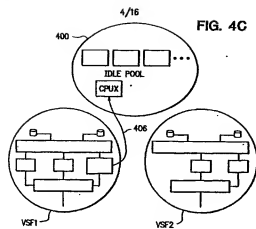
W/O 4183263

PCT/JP04/029653



WO 02/032803

PCT/02/012653



WD 63/6363

PC7/0501/2003

5/16

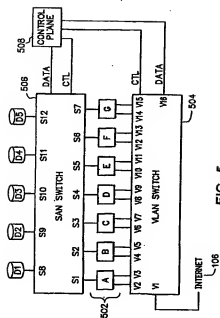


FIG. 5

WO 2004/003

PC/0036/0053

6/16

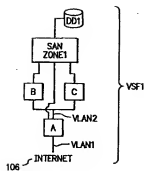


FIG. 6

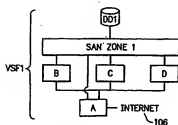


FIG. 7

WD 6263263

PCT/US01/09533

7/16

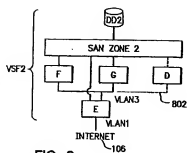


FIG. 8

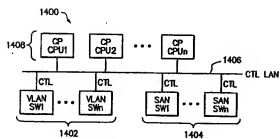
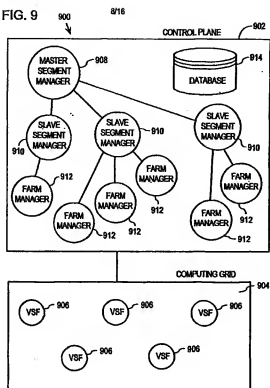


FIG. 14

WO 2004/02953

PC04084/2953

FIG. 9



WO 02/03283

PC200206120653

9/16

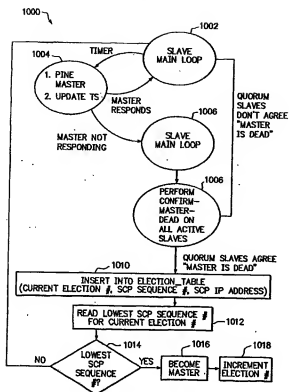
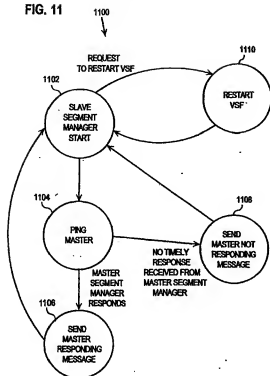


FIG. 10

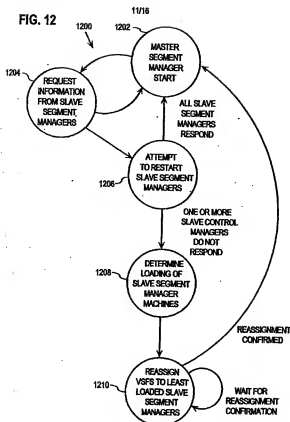
10/16



WFO 82/82003

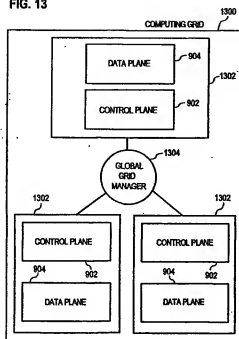
PCT/JP04/02953

FIG. 12



12/16

FIG. 13



13/16

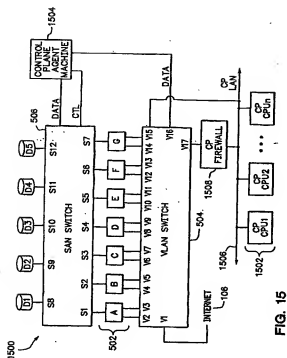


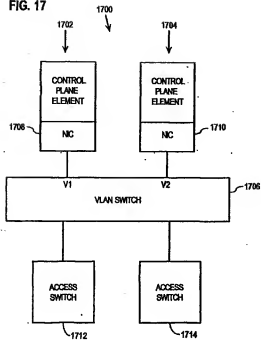
FIG. 15

WO 02/02201

PCT/JP02/02201

14/16

FIG. 17



WFO 42/03203

PCT/JP04/05953

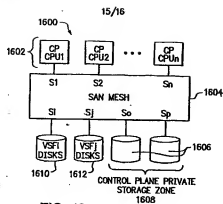


FIG. 16

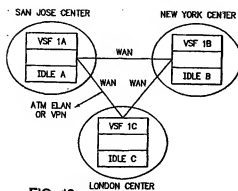
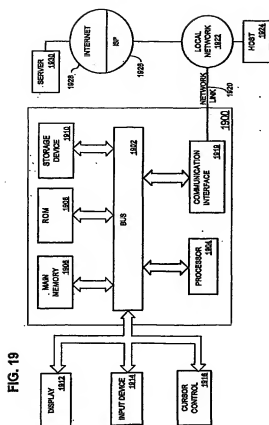


FIG. 18

WO 03/03263

PC/REDA/DMS3

16/16



## 【国際公開パンフレット（コレクトバージョン）】

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization  
International Bureau(43) International Publication Date  
18 January 2002 (18.01.2002)

PCT

(50) International Publication Number  
WO 02/003203 A3(51) International Patent Classification<sup>1</sup>

G06F 9/46

(52) International Application Number: PCT/US99/09853

(53) International Filing Date: 13 Jan 99 (13.01.1999)

(54) Filing Language: English

(55) Publication Language: English

(56) Priority Data:  
60/211,580 30 June 1999 (30.06.1999) US  
09/530,440 2 August 1999 (02.08.1999) US(57) Applicant: TERRASPRING, INC. (US01); 48800 A&S-  
mont Drive, Fremont, CA 94538 (US)(58) Designated States (national): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CZ, DE, DK, EE, ES, FI, GB, GR, HU, IL, IN, JP, KE, KG, KP, KR, KZ, LA, LK, LR, LS, LU, LV, MA, MD, ME, MG, MK, MN, MW, MX, MY, NZ, NI, NO, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, SM, SN, SV, TH, TM, TR, TT, UA, US, UZ, VN, YU, ZA, ZW.  
(59) Designated States (regional): ARIPO patent (GB, GM, KE, LS, MW, MT, MU, NI, SE, TG, TN, TZ, ZM, ZW), European patent (AM, AT, AZ, BA, BB, BG, BR, BY, CH, CN, CZ, DE, DK, EE, ES, FI, GB, GR, HU, IL, IN, JP, KE, KG, KP, KR, KZ, LA, LK, LR, LS, LU, LV, MA, MD, ME, MG, MK, MN, MW, MX, MY, NZ, NI, NO, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, SM, SN, SV, TH, TM, TR, TT, UA, US, UZ, VN, YU, ZA, ZW).Published:  
with international search report

(72) Inventors: AJAY, Ashut; 4180 Tanager Common, Redwood, CA 94555 (US); HARRISON, Brent; 39 Monte Road, San Jose, CA 94085 (US); PATTERSON, Martin; 1445 Honey Street, Mountain View, CA 94041 (US); GRAY, Mark; 664 Palomar Avenue, Mountain View, CA 94041 (US).

(73) Date of publication of the international search report:  
3 April 2002

For two-letter codes and other abbreviations, refer to the "Table over Home on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(74) Agents: BECKER, Edward et al.; Robinson Silverman &amp; Becker, LLP; 1800 Wilcox Street, San Jose, CA 95122 (US).

WO 02/003203 A3

(54) Title: METHOD AND APPARATUS FOR CONTROLLING AN EXTINGUISHABLE COMPUTING SYSTEM

(57) Abstract: Methods and apparatus providing, controlling and managing a dynamically sized, highly available and available server farm are disclosed. A Virtual Server (VSR) is created out of a wide scale computing fabric ("Computing Grid") which is physically constituted from one or more logically divided up host VSRs for various organizations on demand. Each organization creates independent administration control of a VSR. A VSR is dynamically located within the Computing Grid. Allocation and control of the resources in the VSR is performed by a control plane connected to all computing, networking, and storage elements in the computing grid through control plane. The instant topology of each VSR is under control of the control plane. No physical wiring is necessary in order to connect VSRs in many different configurations, including single-tier, multi-tier or multi-tier Web-server, application server, database server configurations.

## INTERNATIONAL SEARCH REPORT

From **RTT DATA** received about 1 July 1988



## INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No.  
PCT/US 01/19053

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0917056 A	19-05-1999	US 2002012850 A1	14-03-2002
		US 6260068 B1	10-07-2001
		US 2002052914 A1	02-05-2002
		US 6260734 B1	01-05-2001
		US 2002016892 A1	07-02-2002
		US 2002010844 A1	24-01-2002
		US 2002016891 A1	07-02-2002
		US 6199179 B1	06-03-2001
		US 6332180 B1	18-12-2001
		EP 0917056 A2	19-05-1999
		EP 0917057 A2	19-05-1999
		JP 11316747 A	16-11-1999
		JP 2000132530 A	12-05-2000
US 5574914 A	12-11-1996	NONE	
EP 0262750 A	06-04-1988	CA 1293819 A1	31-12-1991
		CA 1313276 A2	26-01-1993
		CA 8710607 A , B	09-03-1988
		DE 3751616 D1	11-01-1996
		DE 3751616 T2	09-05-1996
		EP 0262750 A2	06-04-1996
		EP 1700667 A1	01-02-1992
		EP 2792849 A2	03-09-1998
		JP 63145667 A	17-06-1988
		KR 9612054 B1	23-09-1996
		WO 8801772 A1	10-03-1988
		US 5390336 A	14-02-1995
		US 5978070 A	02-11-1999
EP 0935200 A	11-08-1999	US 6247077 B1	12-06-2001
		EP 0935200 A1	11-08-1999
		JP 11328135 A	30-11-1999
WO 0029954 A	25-05-2000	EP 1131719 A1	12-09-2001
		WO 0029954 A1	25-05-2000
		US 2002029319 A1	07-03-2002

Ann. Relevance search results (page 100)

フロントページの続き

(51) Int. Cl. <sup>7</sup>

F I

デーマコード (参考)

G 0 6 F 11/20 3 1 0 E

(81) 指定国 AP (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), EA (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), EP (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OA (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG), AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, S D, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW

(72) 発明者 バターソン, マーティン

アメリカ合衆国 カリフォルニア州 9 4 0 4 1 マウンテン ビュー マーシー ストリート  
1 4 4 5

(72) 発明者 グレイ, マーク

アメリカ合衆国 カリフォルニア州 9 4 0 4 1 マウンテン ビュー フェアモント アベニュー  
6 6 4

Fターム (参考) 5B034 BB02 CC01 CC03 DD05 DD07

5B045 BB28 BB42 GG01 JJ26

5B098 AA03 AA10 GA04 GD02 GD03 GD14

(

(